SOC con Wazuh como SIEM

En este proyecto se tratará de montar un SOC en Vmware Workstation, pero toalmente funcional, se irá actualizando poco a poco. Los elementos principales con los que se va a contar son:

- SIEM (Wazuh)
- IDS (Suricata)
- Adversary simulation (Caldera)
- Honeypots
- pFsense (Enrutar, NATear y aislar las distintas redes)

Índice

Planeamiento

Lo primero que se va a planear son las redes y conexiones, tendremos 3 redes:

- LAN: será la que de acceso a la WAN
- SOC: donde tendremos Wazuh, Suricata y Caldera
- SERVERS: donde tendremos el Honeypot

IMPORTANTE

Al diseñar el laboratorio en vmware workstation (linux), no funciona el modo promiscuo para suricata, o al menos yo no lo conseguí hacer funcionar, algo imprescindible para el "sniffing" de paquetes de la red de Honeypot. Para ello se necesitaría o:

- Poner suricata de proxy entre honeypot y pfsense
- Un switch físico y hacer port-mirroring hacia un interfaz de suricata
- Realizar el laboratorio en workstation Windows (teóricamente funciona)

No obstante, se ha detallado el proyecto para que puedas realizarlo de una de estas 3 maneras. En mi caso, la máquina suricata llevará el honeypot integrado, es decir, Suricata + Honeypot en la misma MV.

La guía se puede seguir perfectamente, a la hora de instalar el honeypot, se avisará de lo que hay que hacer (caso de usar Vmware Workstation en Linux)

**🖧 Configuración de Máquinas Virtuales y Redes **

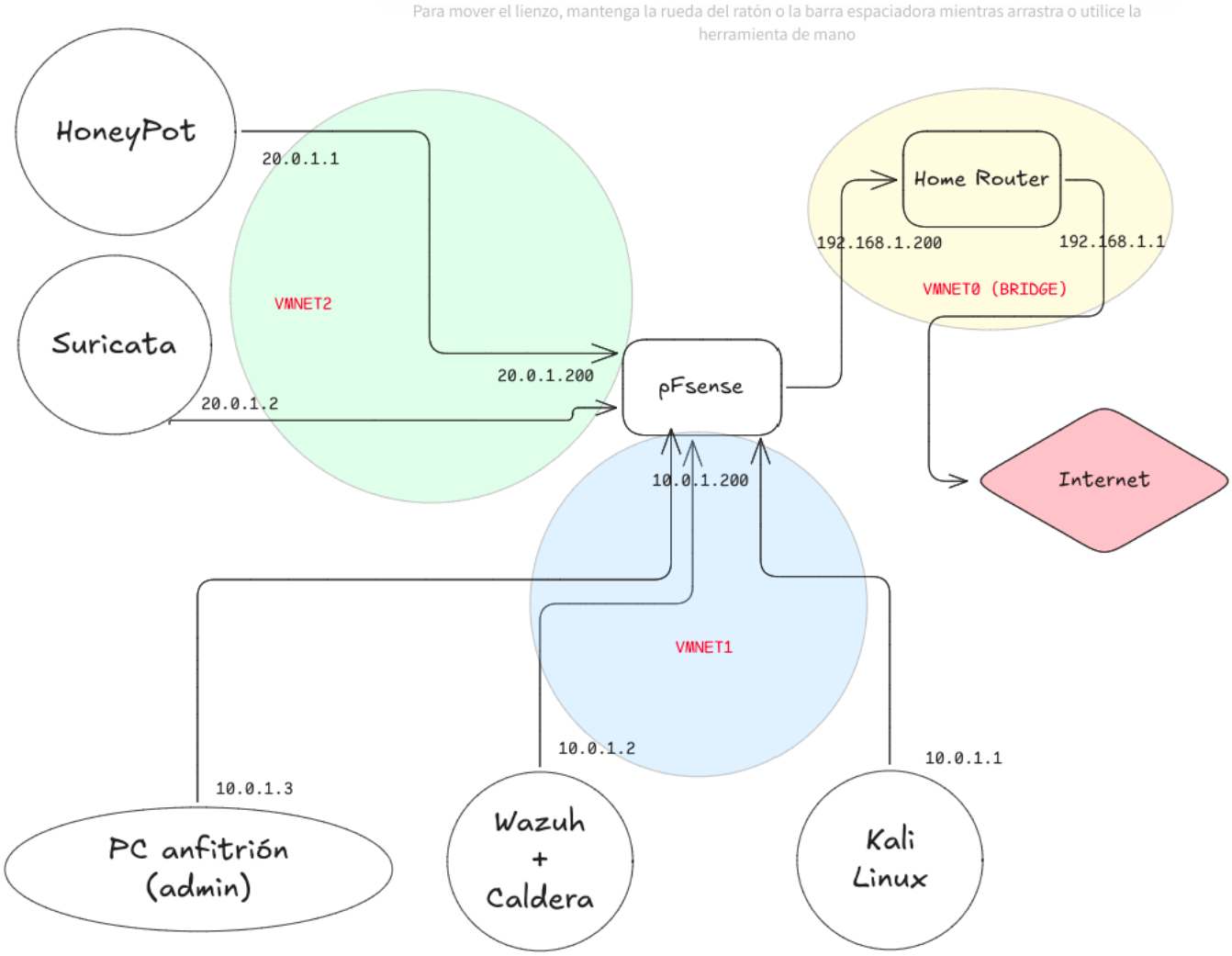
Máquina Virtual	Interfaz	IP	VMnet	Función
pfSense	`WAN (le0)`	**192.168.1.200**	`VMnet0 (Bridge)`	Conexión a Internet
	`LAN1 (le1)`	**10.0.1.200/24**	`VMnet1 (Host-Only)`	Red SOC (Administración pf)
	`LAN2 (le2)`	**20.0.1.200/24**	`VMnet2 (Host-Only)`	Red Honeypot
Suricata	`ens33` (Gestión)	**20.0.1.2/24**	`VMnet2 (Host-Only)`	Red Honeypot (IDS)
Wazuh + Caldera	`ens33`	**10.0.1.2/24**	`VMnet1 (Host-Only)`	Red SOC
Kali Linux	eth0	**10.0.1.1/24**	`VMnet1 (Host-Only)`	Red SOC (Atacante pruebas)
Honeypot	`ens34`	**20.0.1.1/24**	`VMnet2 (Host-Only)`	Red Honeypot (Atacada)

🔑 Tabla de Credenciales del Laboratorio

Máquina/Servicio	Usuario	Contraseña	Notas
pfSense Web GUI	`admin`	`cuTRfyC6OOCui6`	Acceso a la interfaz web solo en la**Red SOC** (`https://10.0.1.200`)

Máquina/Servicio	Usuario	Contraseña	Notas
pfSense SSH	`admin`	`cuTRfyC6OOCui6`	Acceso remoto por SSH solo desde la **Red SOC** (`ssh root@10.0.1.200`)
Honeypot (Cowrie, Dionaea, etc.)	`root`	`honeypot123`	Dependiendo del honeypot instalado
Wazuh Web UI	`admin`	`kpAGnmrQHiymIWRnO?mPEh*L46NJo6.e`	Acceso a la consola Wazuh (`https://10.0.1.1`)
Wazuh SSH	`administrator`	`SOCproyecto.2025!`	Acceso al servidor Wazuh
Suricata SSH	`administrator`	`SOCproyecto.2025!`	Acceso al servidor Suricata (`10.0.1.2`)
Caldera (Red Team Framework)	`admin`	`SOCproyecto.2025!`	Acceso a `https://10.0.1.1:8888`
Kali linux	`kali`	`kali`	

Esquema de red



Requisitos previos

Para no extender mucho esta guía, se omitirán ciertos pasos pero que son requisito previo e indispensable:

- Crear Vmnets
- Asegurarse que el Vmnet0 en modo bridge está en bridge con la interfaz deseada
- Configurar router de la red de casa para:
    - pFsense (192.168.1.200) tenga todos sus puertos abiertos. Se puede meter en DMZ
    - Reenvío de tráfico de los servicios que queramos ser atacados hacia el pFsense
- Configurar el equipo anfitrión en la red de SOC para poder acceder a las interfaces de wazuh, caldera, etc.

#Windows Tan sencillo como editar la interfaz desde el administrador de interfaces

#Linux (Debian 12) Crear un fichero para el interfaz: sudo nano /etc/systemd/network/10-vmnet1.network (Para otras ocasiones solo cambiar el vmnet, lo del "10-" se mantiene).

```
[Match]
Name=vmnet1

[Network]
Address=10.0.1.3/24
Gateway=10.0.1.200
```

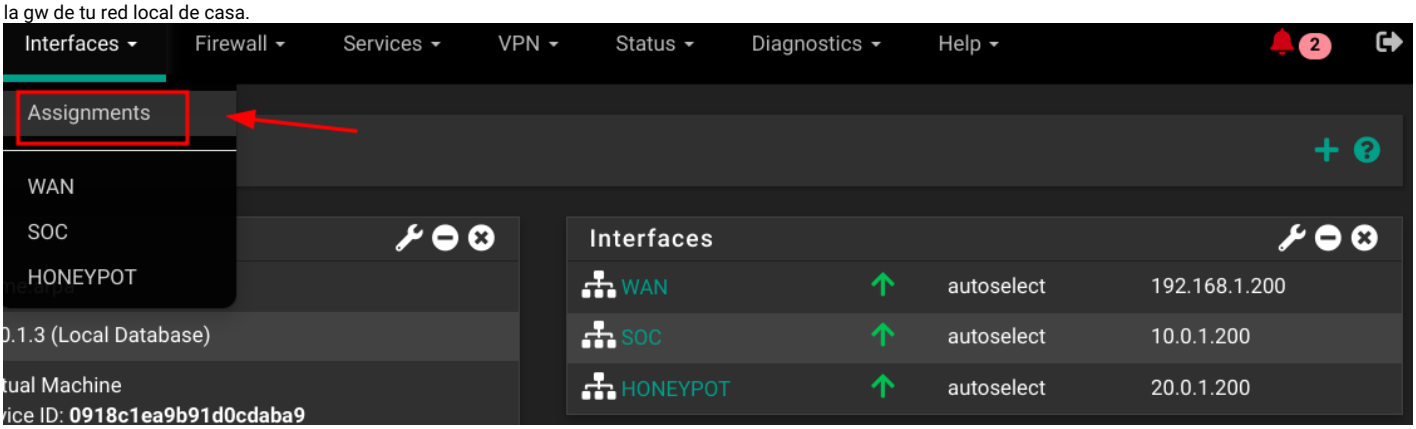
`sudo systemctl restart systemd-networkd`

Instalación y configuración pFsense

Una vez tengamos la MV instalada, le agregaremos 3 interfaces de red conectados a:

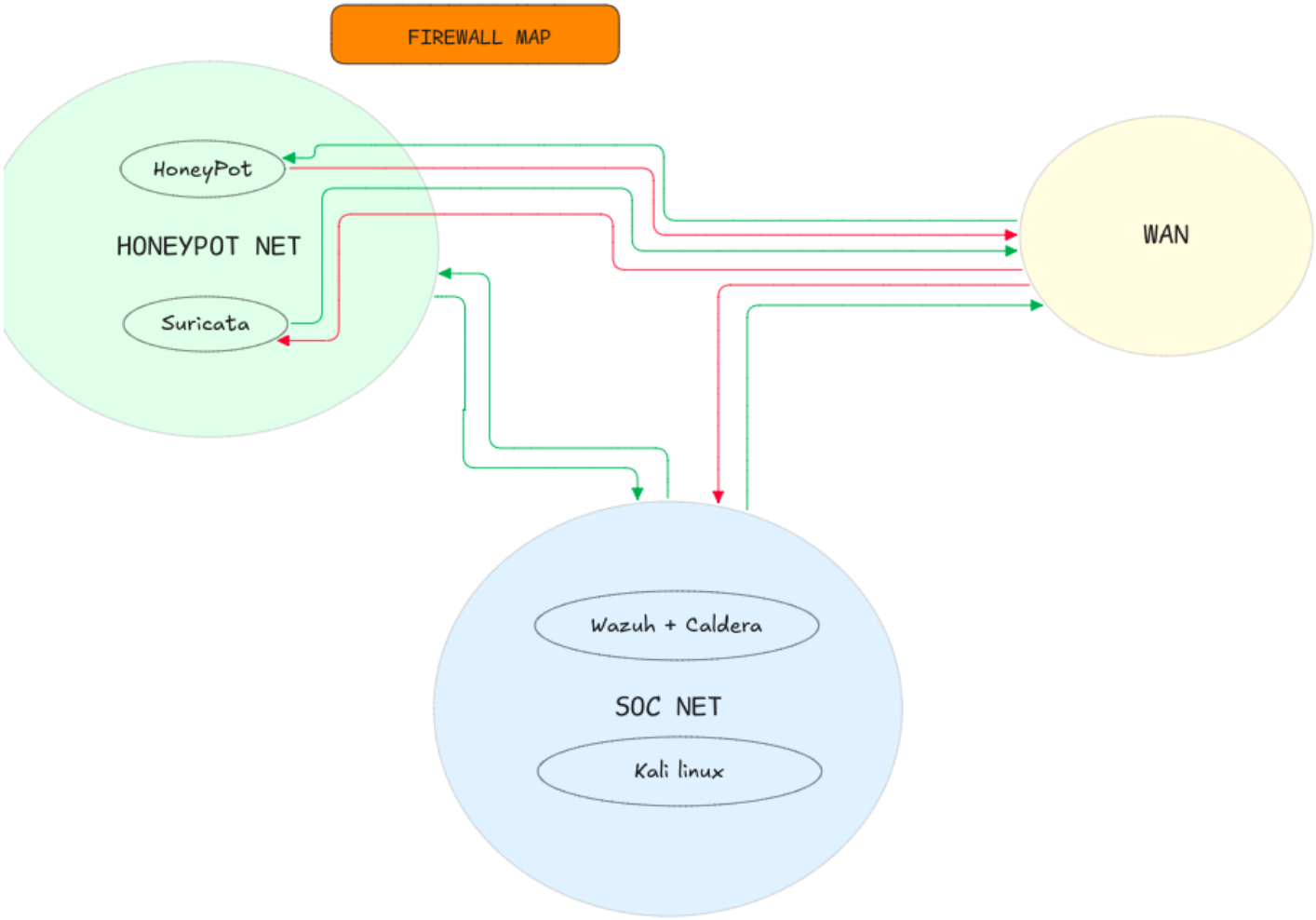
- Vmnet0 será la WAN
- Vmnet1 será SOC
- Vmnet2 será Honeypot

Se habilita el interfaz, asignamos un nombre, establecemos IP estática, cambiamos el CIDR y nos aseguramos que la interfaz de WAN, su gateway de "upstream" sea la gw de tu red local de casa.



El siguiente paso es configurar las reglas de firewall, las principal intención de comunicación es:

- Honeypot puede ser accedido dese WAN (para ser atacado)
- Honeypot puede acceder a SOC (para enviar logs a Wazuh)
- Honeypot puede ser accedido desde SOC (Para acceder desde Kali, Wazuh, Caldera, etc)
- Suricata NO puede ser accedido desde WAN (no tiene motivos para ello)
- Suricata puede acceder a WAN (buscar updates)
- Suricata puede acceder a SOC (enviar logs a Wazuh)
- Suricata puede ser accedido desde SOC
- SOC puede acceder a WAN (updates e internet)
- SOC NO puede ser accedido desde WAN (no tiene motivos para ello)



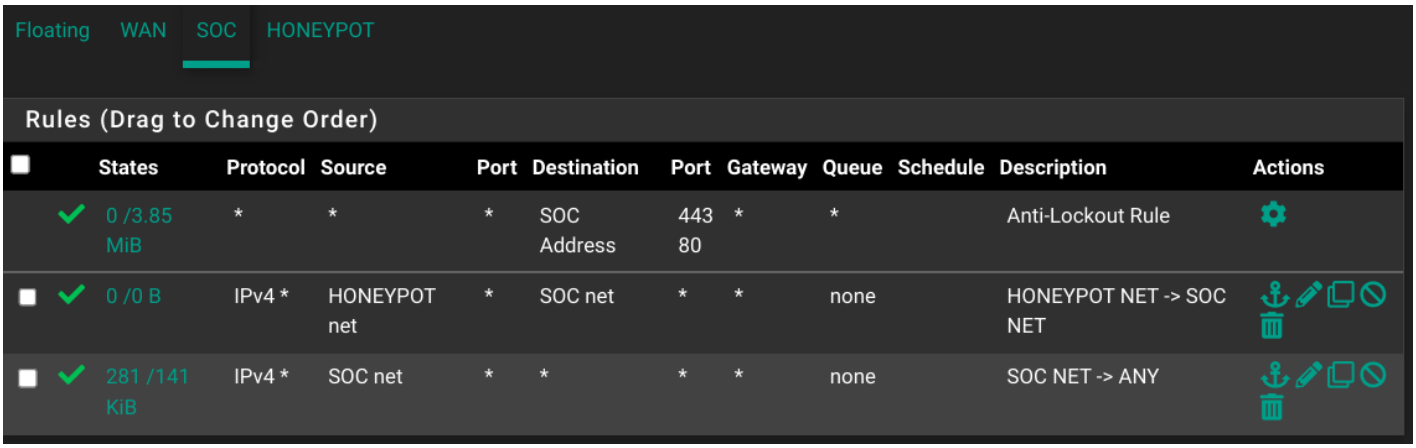
Configuración firewall WAN Interface

Floating | WAN | SOC | HONEYPOT

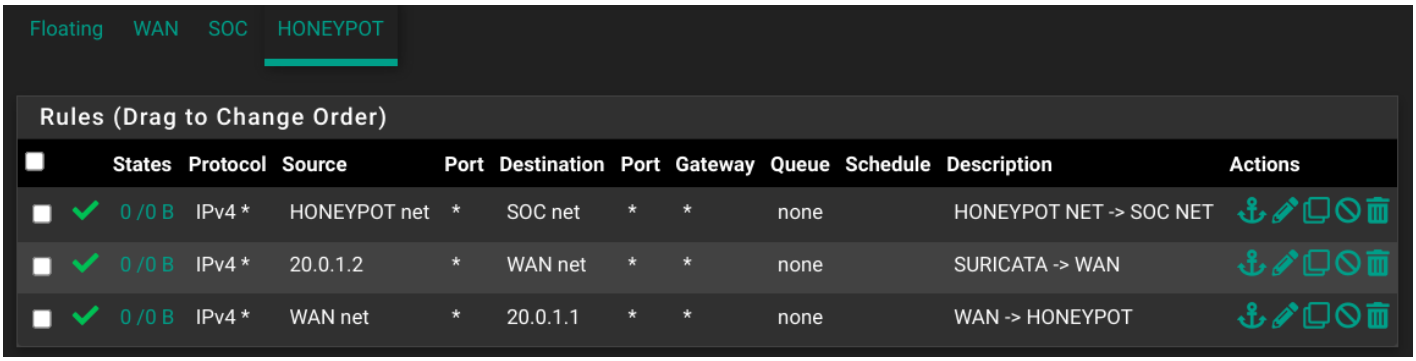
Rules (Drag to Change Order)

	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
✓	0 /0 B	IPv4 *	SOC net	*	WAN net	*	*	none		SOC NET -> WAN	
✗	0 /0 B	IPv4 *	WAN net	*	SOC net	*	*	none		WAN -> SOC NET	
✗	0 /0 B	IPv4 *	WAN address	*	20.0.1.2	*	*	none		WAN -> SURICATA	
✓	0 /0 B	IPv4 *	WAN net	*	20.0.1.1	*	*	none		WAN -> HONEYPOT	
✗	0 /0 B	IPv4 *	20.0.1.1	*	*	*	*	none		HONEYPOT -> WAN	
✓	0 /0 B	IPv4 *	20.0.1.2	*	*	*	*	none		SURICATA - > WAN	

Configuración firewall SOC Interface



Configuración firewall HONEYPOT Interface



Instalación y configuración Wazuh

Instalamos (En mi caso Ubuntu Server 22.04), asignamos Vmnet1, establecemos direccionamiento IP (10.0.1.2/24 - Gw 10.0.1.200).

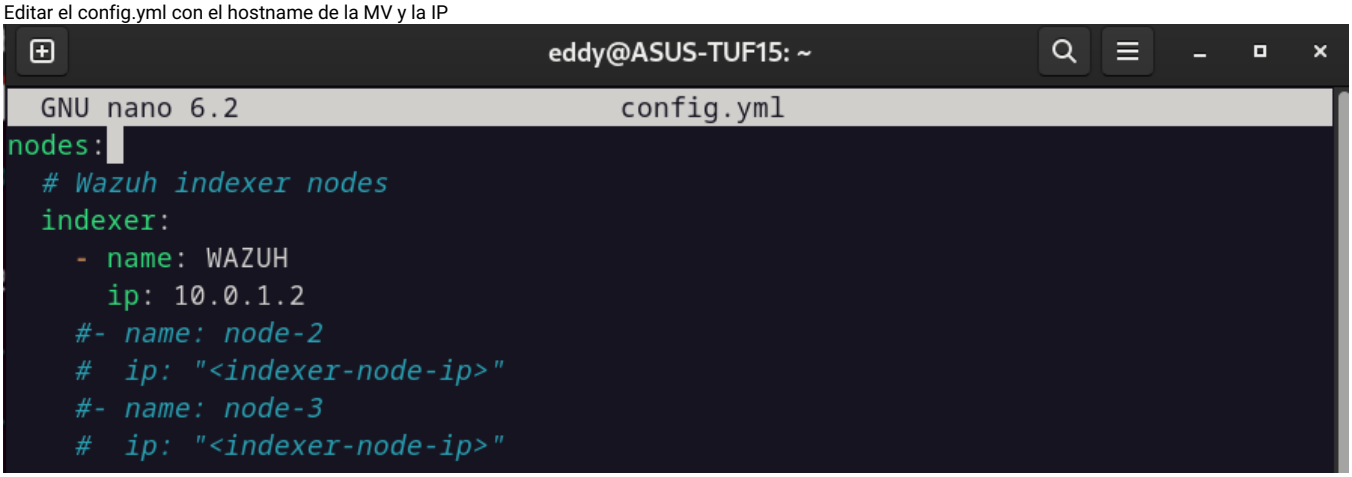
```
apt update && apt upgrade -y
```

NOTA:Dependiendo de la versión de wazuh, los comandos pueden variar, lo mejor es consultar la documentación oficial

15. Descargar el script de instalación (la URL variará)

```
curl -sO https://packages.wazuh.com/4.10/wazuh-install.sh
curl -sO https://packages.wazuh.com/4.10/config.yml
```

16. Editar el config.yml con el hostname de la MV y la IP



17. Generar ficheros de configuración e instalar

```
bash wazuh-install.sh --generate-config-files
bash wazuh-install.sh -a
```

18. Eliminar repositorios de wazuh, para evitar un upgrade accidental que pueda estropear el servicio. (OPCIONAL)

```
sed -i "s/^deb /#deb /" /etc/apt/sources.list.d/wazuh.list
apt update
```

Instalación y configuración Suricata

Instalamos el S.O, asignamos 2 interfaces de red al Vmnet1, y establecemos direccionamiento IP.

```
apt update && apt upgrade -y
add-apt-repository ppa:oisf/suricata-stable
apt install suricata -y



# Descargar emmerging rules
cd /tmp/ && curl -LO https://rules.emergingthreats.net/open/suricata-6.0.8/emerging.rules.tar.gz
sudo tar -xvzf emerging.rules.tar.gz && mkdir /etc/suricata/rules && sudo mv rules/*.rules /etc/suricata/rules/
    sudo chmod 640 /etc/suricata/rules/*.rules
```

19. Editar configuración para especificar la interfaz de red, y la HOME_NET (la red que queremos monitorear), ademas de las nuevas rules descargadas.

```
/etc/suricata/suricata.yml

# buscar esta linea y editar tal que así
af-packet:
  - interface: ens33

#buscar la esta linea y editar con las redes que queremos monitorear, en este caso la del soc y la de honeypot
vars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[20.0.1.0/24]"

# buscar default-rule y editar tal que asi
default-rule-path: /etc/suricata/rules
rule-files:
- "*.rules"
```

20. Habilitar al inicio e iniciar

```
systemctl enable suricata
systemctl start suricata
```

21. Comprobar alertas

```
#Ya deberiamos ver los scans de nmap, por ejemplo
tail -f /var/log/suricata/fast.log
```

Envío logs Suricata > Wazuh

Para ello necesitaremos tener un agente de Wazuh instalado en suricata. En el panel de Wazuh > Deploy Agents. Rellenamos con la IP del servidor, elegimos S.O destino, etc. y nos generará un comando similar al siguiente, que descargará el agente de los servidores de Wazuh (internet) con la configuración deseada. En caso de ser offilne, se puede descargar e importar el agente de manera offline.

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.10.1-1_amd64.deb && sudo WAZUH_MANAGER='10.0.1.2' dpkg
```

Editamos el siguiente fichero para decirle al agente de wazuh donde están los logs: ***Si no funciona, agregar tambien al fichero ossec.conf del manager, desde la interfaz web***

```
/var/ossec/etc/ossec.conf


# eal final del fichero en el ultimo <ossec_config> (no en el primero)

    <localfile>
        <log_format>syslog</log_format>
        <location>/var/log/suricata/eve.json</location>
    </localfile>
```

Por último habilitamos el servicio y lo iniciamos:

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

Instalación y configuración Honeypot

Llegados a este punto, si virtualizas en workstation con linux, debes agregar una regla para permitir el tráfico desde WAN hacia suricata. Si tu caso es cualquier otro descrito en esta guía, puedes seguir normalmnete., teniendo en cuenta en que MV instalar el Honeypot.

Instalación

```
apt
apt install python3-pip
pip install honeypots
```

Lanzar honeypot

Honeypots tiene una larga lista de servicios que se pueden lanzar, para ver el listado:

Listar honeypots

```
honeypots --list
```

Lanzar honeypot

```


# Ejepmplo lanazr SSH
python3 -m honeypots --setup ssh --auto

# Lanzar todos:
python3 -m honeypots --setup all --auto
```

Instalación e inicio Caldera

Caldera es un servicio para la emulación del adversario, de esta manera podremos lanzar ataques simulados a nuestro honeypot y comprobar la detección por parte de wazuh.

```
git clone https://github.com/mitre/caldera.git --recursive
cd caldera
pip3 install -r requirements.txt

#### INICIAR EL SERVICIO
python3 server.py --insecure --build
```

Instalación docker (alternativa)

```
# Recursively clone the Caldera repository if you have not done so
git clone https://github.com/mitre/caldera.git --recursive

# Build the docker image. Change image tagging as desired.
# WIN_BUILD is set to true to allow Caldera installation to compile windows-based agents.
# Alternatively, you can use the docker compose YML file via "docker-compose build"
cd caldera
docker build . --build-arg WIN_BUILD=true -t caldera:latest

# Run the image. Change port forwarding configuration as desired.
docker run -p 8888:8888 caldera:latest
```

Tras el despliegue, debemos desplegar un agente en el honeypot, para lanzar los ataques simulados y ver si en wazuh vemos los detalles de esos ataques.

ACCESO

`https://<ip>:8888`

File Integrity Monitoring (FIM)

Trata de monitorizar la integridad de ciertos ficheros y directorios importantes. para ello en el agente de la MV en cuestión (honeypot), debemos agregar los directorios y/o ficheros que queramos monitorizar.

Dependiendo de la maquina, en un entorno real, unos directorios serán importantes y otros no. Debemos conocer nuestro servicio y la criticidad para saber determinar que es importante y que no.

```
## EQUIPO CLIENTE LINUX CON AGENTE WSUS
/var/ossec/etc/ossec.conf

# Apartado de File integrity monitoring
<syscheck>
  <disabled>no</disabled>
  <frequency>720</frequency>
  <scan_on_start>yes</scan_on_start>
  <directories check_all="yes" report_changes="yes" real_time="yes">/etc,/bin,/sbin</directories>
  <directories check_all="yes" report_changes="yes" real_time="yes">/lib,/lib64,/usr/lib,/usr/lib64</directories>
  <directories check_all="yes" report_changes="yes" real_time="yes">/var/www,/var/log,/var/named</directories>
  <ignore>/etc/mtab</ignore>
  <ignore>/etc/hosts.deny</ignore>
  <ignore>/etc/mail/statistics</ignore>
  <ignore>/etc/random-seed</ignore>
  <ignore>/etc/adjtime</ignore>
  <ignore>/etc/httpd/logs</ignore>
  <ignore>/etc/utmpx</ignore>
  <ignore>/etc/wtmpx</ignore>
  <ignore>/etc/cups/certs</ignore>
  <ignore>/etc/dumpdates</ignore>
  <ignore>/etc/svc/volatile</ignore>
  <ignore>/sys/kernel/security</ignore>
  <ignore>/sys/kernel/debug</ignore>
  <ignore>/sys</ignore>
  <ignore>/dev</ignore>
  <ignore>/tmp</ignore>
  <ignore>/proc</ignore>
  <ignore>/var/run</ignore>
  <ignore>/var/lock</ignore>
  <ignore>/var/run/utmp</ignore>
</syscheck>
```

Custom SCA con Wazuh

Cumplimiento CCN-STICS

El **Security Configuration Assessment (SCA)** es una herramienta clave para evaluar el cumplimiento de estándares de seguridad como **CIS** o **NIST**, verificando configuraciones como el tiempo de desconexión tras inactividad.

En este caso se desarrolla el cumplimiento de las **CCN-STICS** para cumplir con el marco del **ENS** (Esquema nacional de seguridad Español)

Debido a lo extenso que es este apartado, se ha desarrollado en otro artículo, publicado en la siguiente url: Custom SCA con Wazuh

CONTINUARÁ EL DESARROLLO CON:

- CDB lists
- VirusTotal integration
- Windows defender logs integration
- Sysmon integration
- TheHive + Cortex
- MISP
- Shuffle SOAR
- Active response scripts
- Threat Hunting