PRACTICA BLUE TEAM

CONTENIDO:

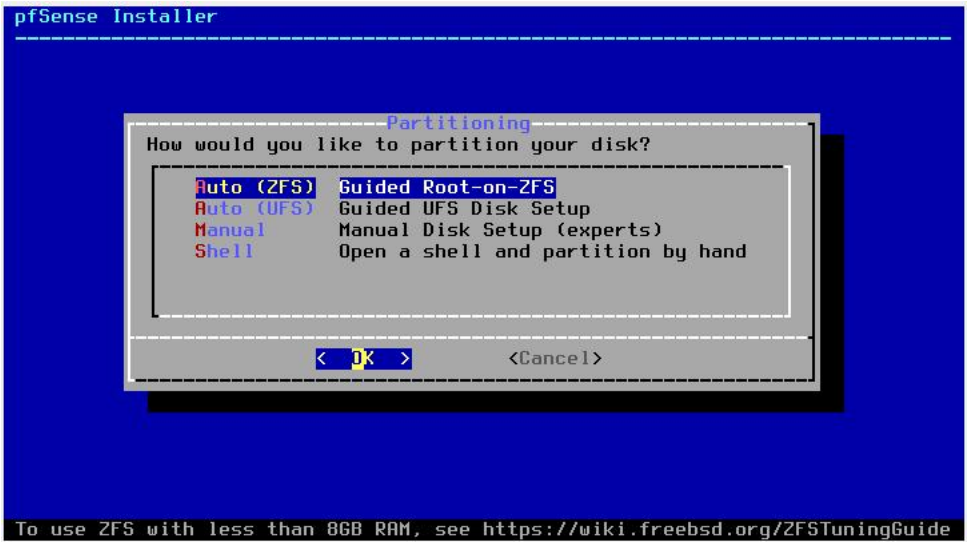
1. **PFSENSE**:

• SO: free BSD
• Memoria: 1024MB
• CPUs:1
• Almacenamiento: 20GB

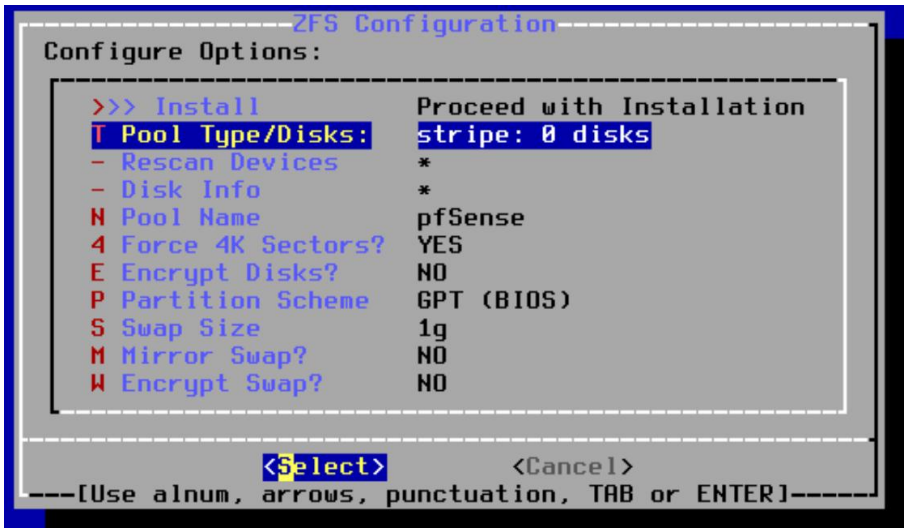
1.1 **INSTALACION DEL PFSENSE**:

- Intalamos:

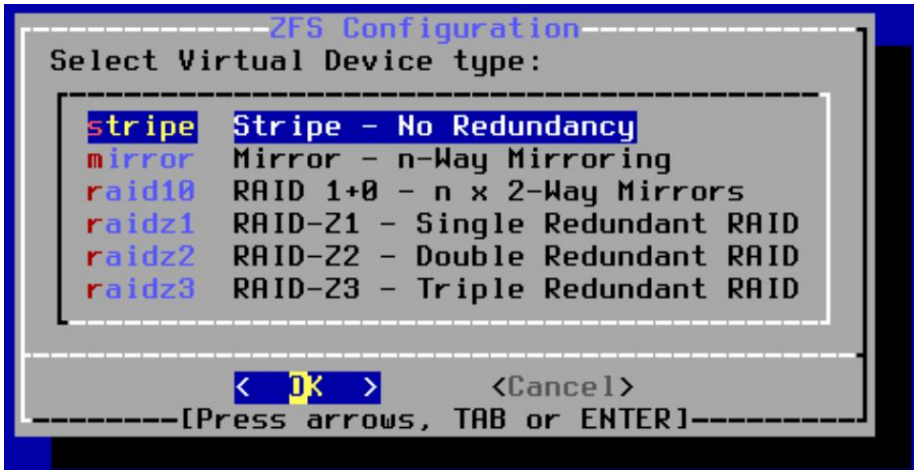
- Le damos a Guided Root-on-ZFS



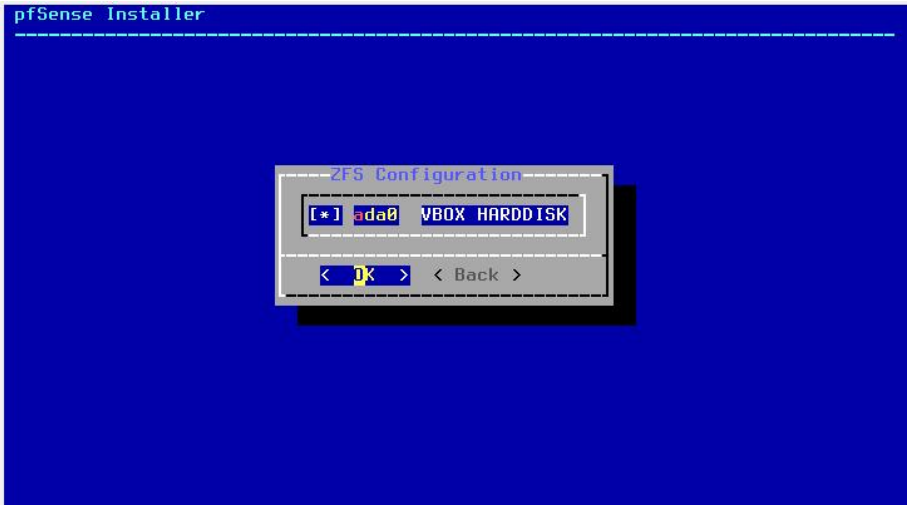
- Marcamos Pool Type/Disks



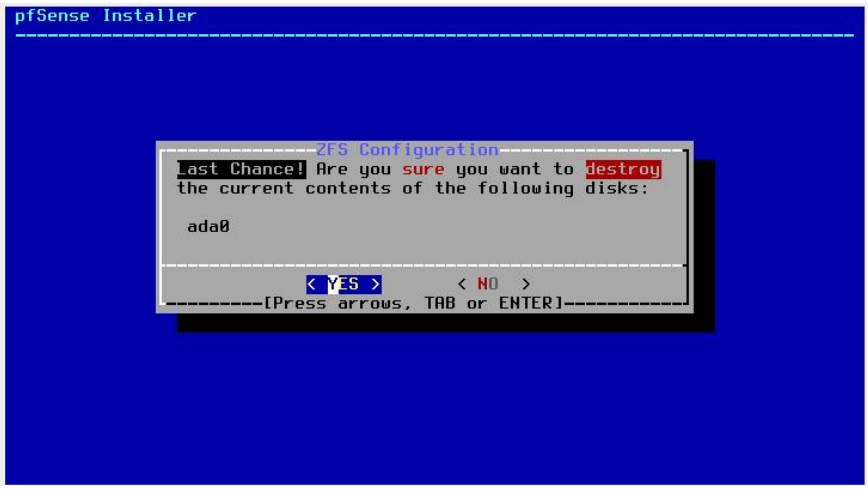
- Ahora vamos a Stripe Stripe – No Redundancy



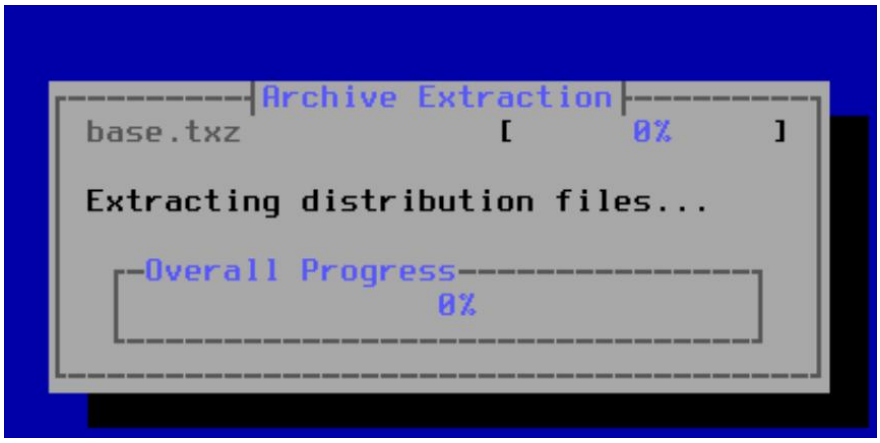
- Y marcamos con la barra espaciadora ada0 VBOX HARDDISK



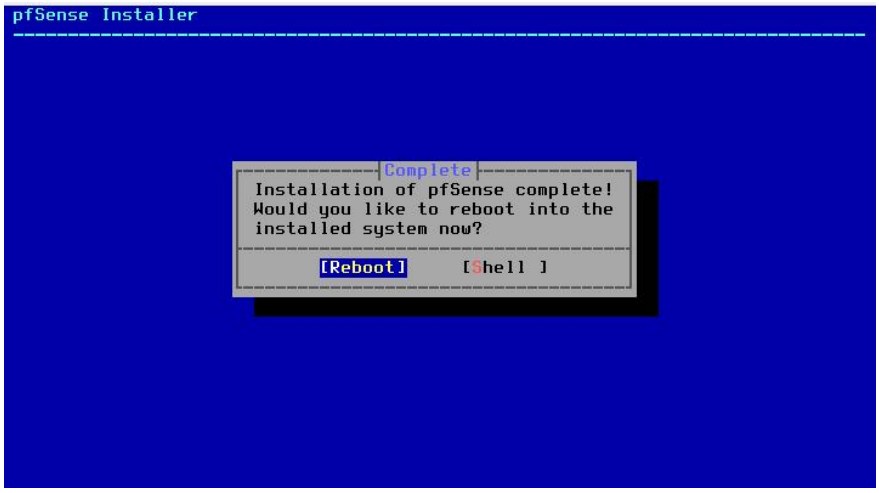
- Le damos a instalar y aquí marcamos Destroy data



- Comienza la instalación



- Una vez instalada, le damos a reboot

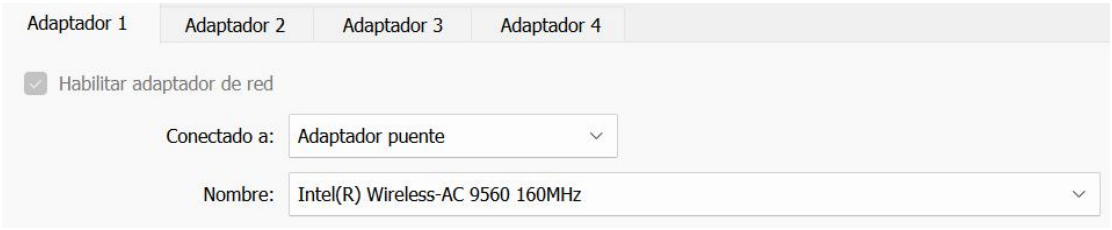


1.2CREACIÓN DE LAS REDES EN VIRTUALBOX

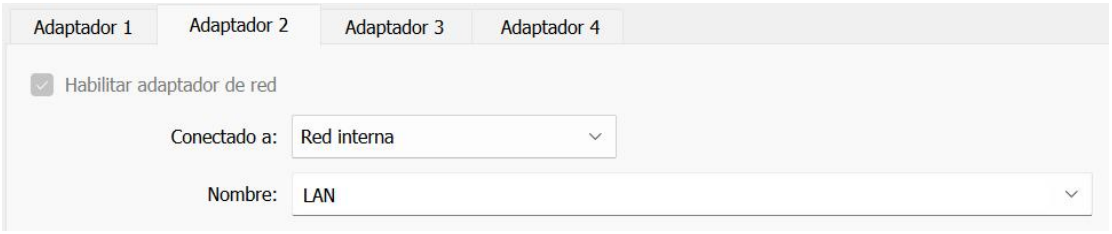
Lo primero que vamos a hacer es la creacion de las redes de nuestro UTM.

Para ello, en Virtual Box, vamos a la configuración de nuestro UTM y lo configuramos de la siguiente manera:

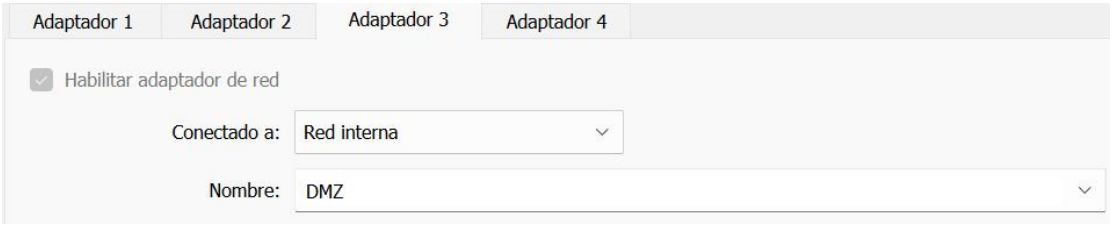
Adaptador 1:



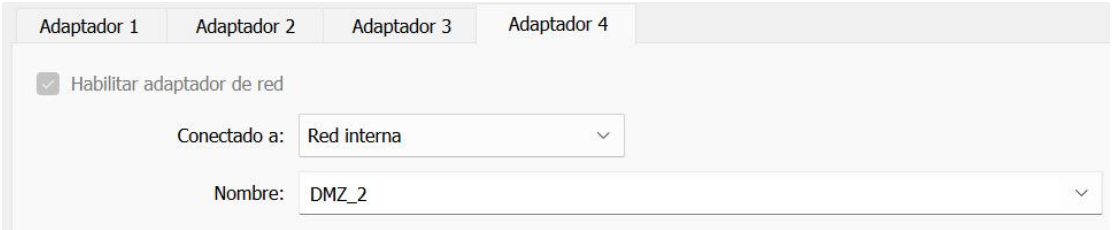
Adaptador 2-LAN:



Adaptador 3-DMZ:

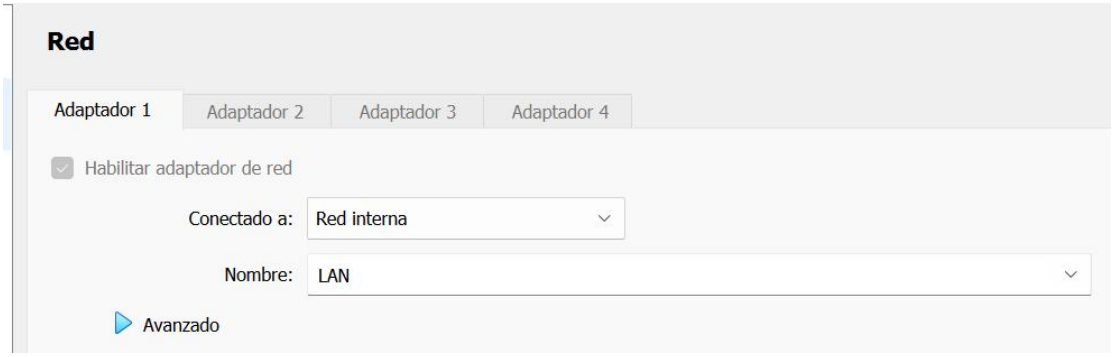


Adaptador 4-DMZ_2

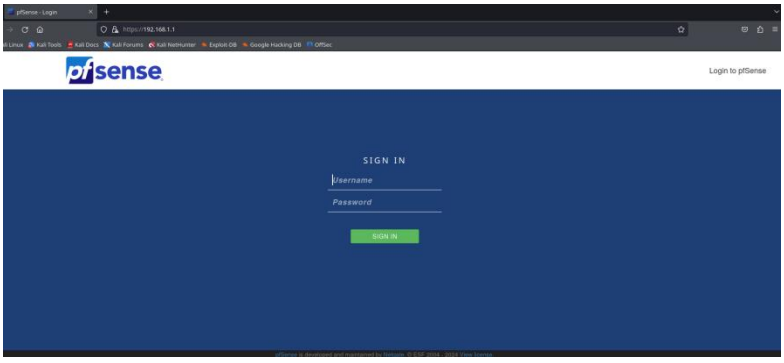


Si ahora arrancamos el UTM deberíamos ver esto:

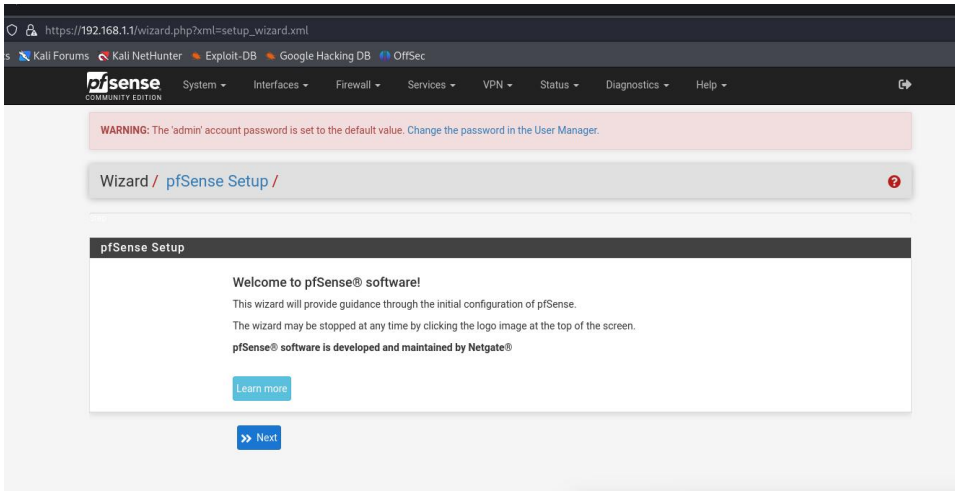
Ahora vamos a añadir nuestra maquina Kali a la LAN para configurar el Pfsense.



Una vez ya tenemos las redes configuradas, ya podemos acceder a nuestro Pfsense a través de la dirección ip: 192.168.1.1 desde el navegador de nuestro kali linux.

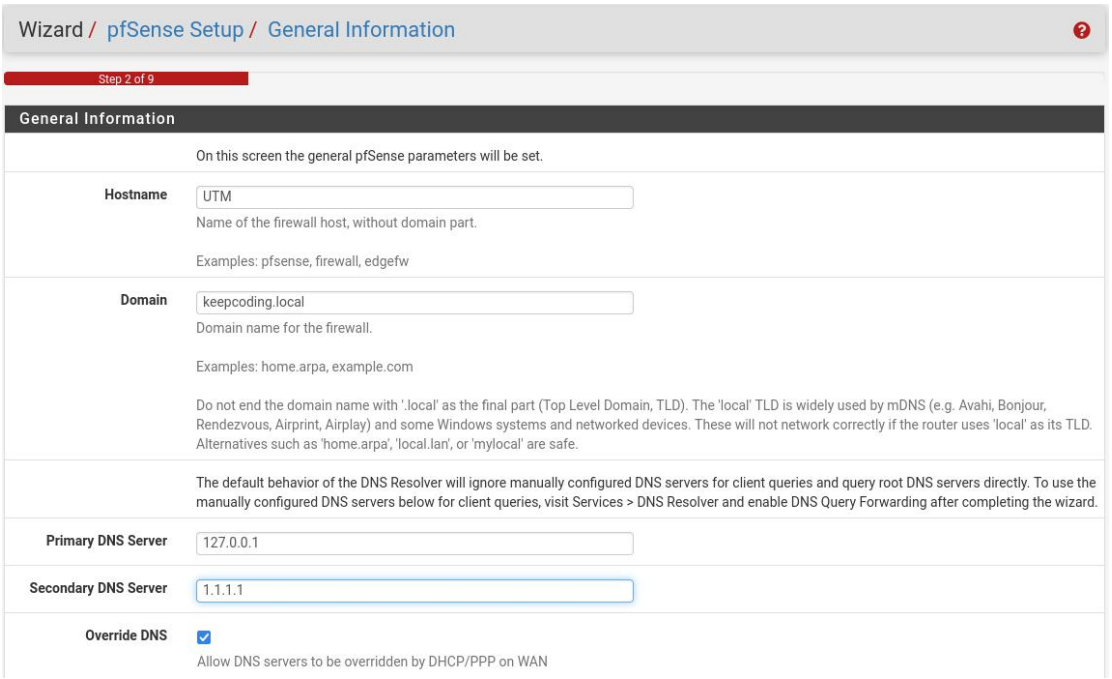


Nos logeamos con el **nombre**: admin. **contraseña**: pfsense, y ya estamos dentro para empezar con la configuración.



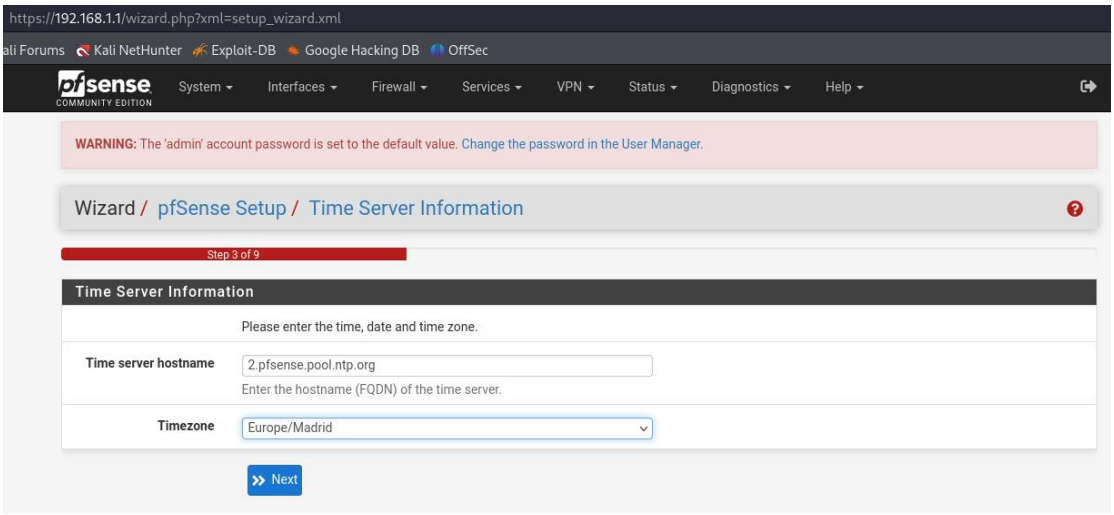
Vamos a System>setup wizzard rellenamos la configuración:

Nombre del host: UTM
Nombre de dominio: keepcoding.local.
Servidor dns primario: 127.0.0.1 (para que sea el propio pfsense el primer servidor dns).
Servidor dns secundario: 1.1.1.1. (este servidor dns son los servidores de cloudfair)

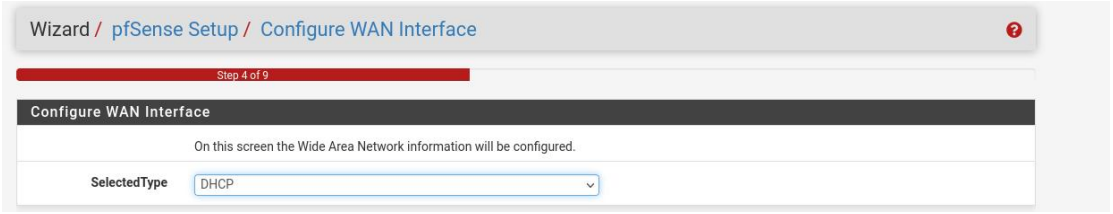


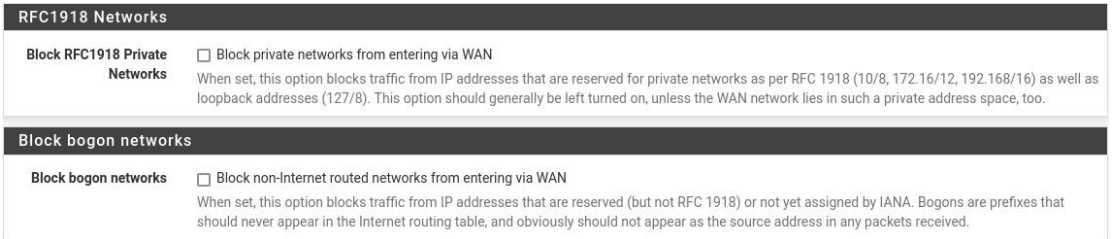
Aquí,en Time server information lo configuramos de la siguiente manera:

Time server hostname: 2.pfsense.pool.ntp.org
Timezone: Europe/Madrid



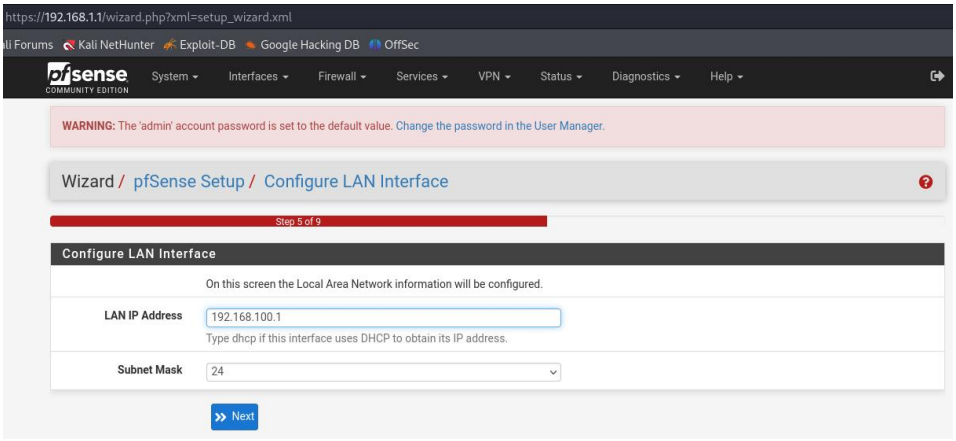
Dejamos todo en modo DHCP ya que va a ser el PfSense el que actúa como servidor DHCP y permitimos el uso de IPs privadas (RFC1918 y Bogons), para que no nos bloquee las IPs internas que vamos a utilizar.



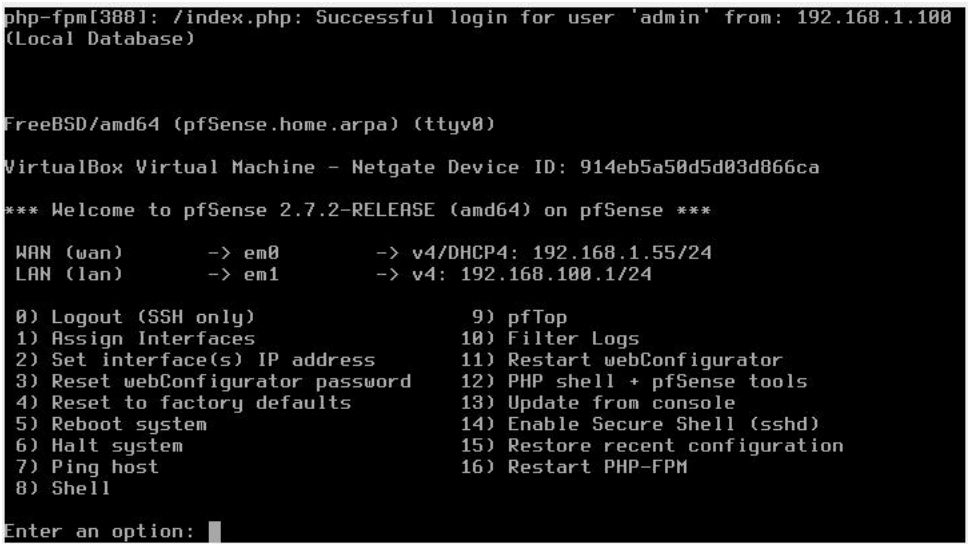


Aquí, en la configuracion de LAN, vamos a cambiar el rango de direcciones, ahora mismo la direcciones que tenemos son las de ifconfig de nuestro kali linux y lo que vamos a hacer es modificarla para que en lugar de ser esta la puerta de enlace 192.168.1.1 sea 192.168.100.1, para que no entre en conflicto:

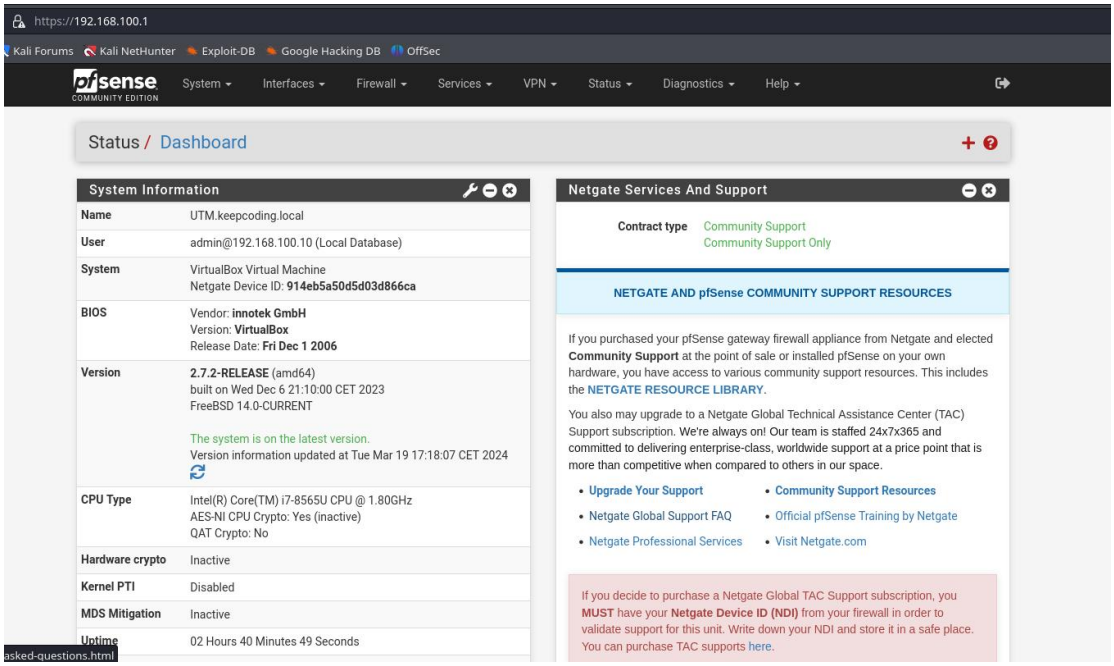
LAN IP address: 192.168.100.1
Subnet: Mask: 24



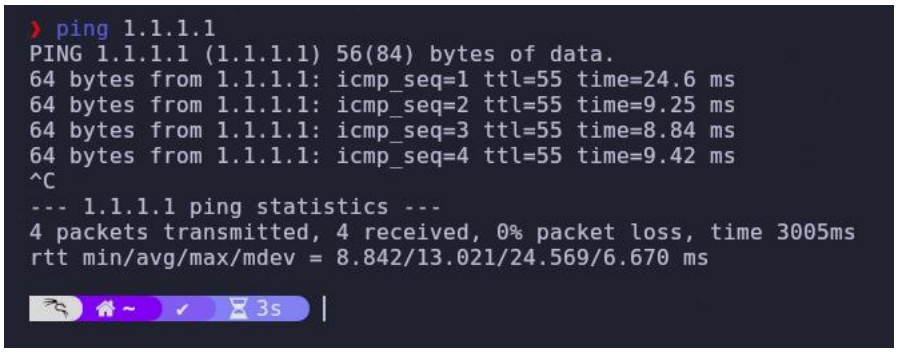
Ahora mismo si vamos al UTM deberíamos ver si le damos a intro que la dirección de nuestra puerta de enlace LAN ha cambiado a la nueva que hemos establecido



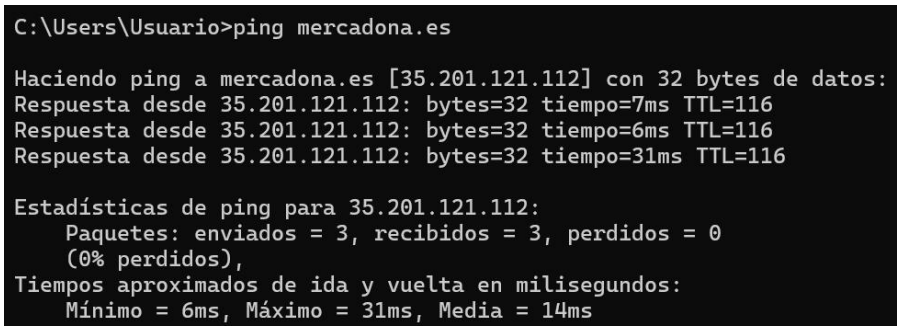
Para poder acceder al pfsense ponemos la nueva dirección IP que hemos configurado.



En primer lugar comprobamos si tenemos acceso a internet haciendo un ping 1.1.1.1, y vemos que si responde, pero en cambio, si hacemos un ping a mercadona.es no responde



Vamos a nuestro cmd de windows y vamos a ver cual es la ip de mercadona.es hacemos ping mercadona.es y nos sale 35.201.121.112



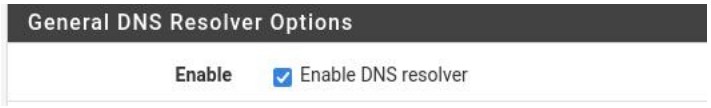
Por lo tanto, vemos que por tema de resolución de dominio no llegamos, pero poniendo la IP en nuestra kali si llegamos, por lo tanto vemos que tenemos un problema de resolución dns.

```
❯ ping 35.201.121.112
PING 35.201.121.112 (35.201.121.112) 56(84) bytes of data.
64 bytes from 35.201.121.112: icmp_seq=1 ttl=115 time=7.48 ms
64 bytes from 35.201.121.112: icmp_seq=2 ttl=115 time=8.37 ms
64 bytes from 35.201.121.112: icmp_seq=3 ttl=115 time=9.07 ms
64 bytes from 35.201.121.112: icmp_seq=4 ttl=115 time=8.77 ms
^C
--- 35.201.121.112 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 7.481/8.423/9.067/0.596 ms
```

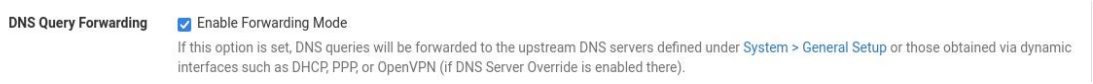
Ahora vamos a **configurar la resolución DNS**.

Para ello vamos a Services-DNS resolver

- Comprobamos que está habilitado
- Quitamos el **DNSSec**.Es un protocolo que se utiliza para el firmado de las respuestas DNS, cuando haces una consulta DNS la consulta puede ir firmada para que se indique que nadie te la ha modificado por el camino (la consulta que te devuelva, te la envíe a ti y puedes confirmar esa firma para ver que no es un DNS fake)



- Habilitamos el **Forwarding mode** para que pueda realizar las consultas DNS en caso de que pfsense no sea capaz por si mismo, en este caso lo que haria seria fowardearlas al DNS secundario que seria el 1.1.1.1. de cloudfair

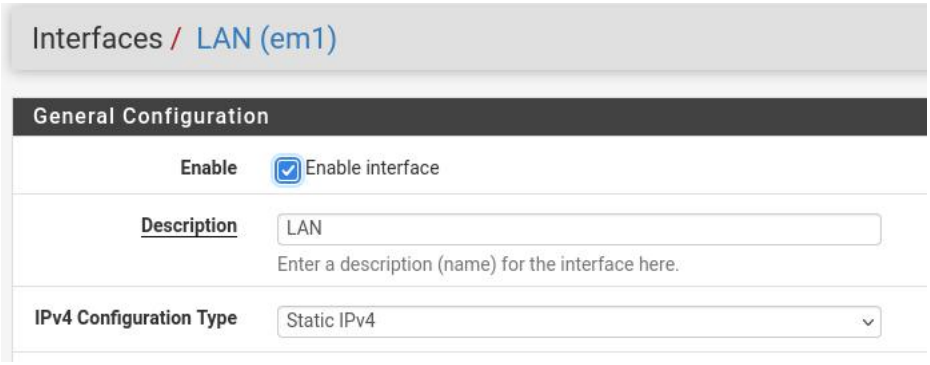


Con esto ya tendríamos la configuración DNS. lanzamos otra vez el ping a mercadona.es para comprobarlo.

```
❯ ping mercadona.es
PING mercadona.es (35.201.121.112) 56(84) bytes of data.
64 bytes from 112.121.201.35.bc.googleusercontent.com (35.201.121.112): icmp_seq=1 ttl=115 time=105 ms
64 bytes from 112.121.201.35.bc.googleusercontent.com (35.201.121.112): icmp_seq=2 ttl=115 time=9.02 ms
64 bytes from 112.121.201.35.bc.googleusercontent.com (35.201.121.112): icmp_seq=3 ttl=115 time=29.1 ms
^C
--- mercadona.es ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 9.018/47.715/105.066/41.370 ms
```

Ya tenemos nuestro DNS funcionando y configurado, ahora queremos poner un rango de direccionamiento a cada una de las redes, para ello nos vamos a configurar para que el servidor DHCP de nuestra red nos otorgue ese rango en las maquinas que tenemos en esa subred.

Para ello vamos a interfaces y le damos a LAN, aquí deberíamos tener la interfaz habilitada y en static ipv4.

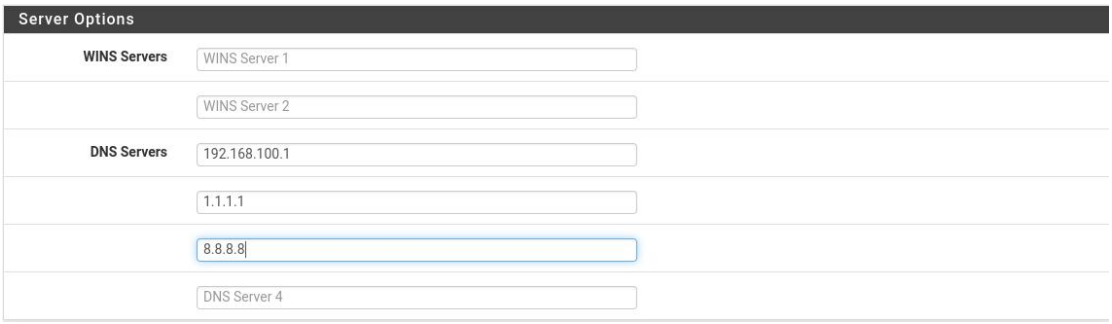


Ahora vamos en service a DHCP-SERVER-LAN.

- Modificamos el rango de direcciones de la LAN (192.168.100.100 to 192.168.100.200) con el fin de conseguir la subred que tenemos definido en nuestro esquema de red.

- **Modificamos los servidores DNS**, ponemos el propio pfsense (hay que tener en cuenta que hay que poner la IP de puerta de enlace, por la que vamos a acceder a nuestra maquina kali, que sería 192.168.100.1).

- En la red LAN y como **servidores DNS secundarios** ponemos el 1.1.1.1 (cloudfair) y el 8.8.8.8 (google), por si el DNS no fuese capaz de hacer la resolución de nombres.



- Modificamos la puerta de enlace/Gateway, ponemos la misma dirección, 192.168.100.1

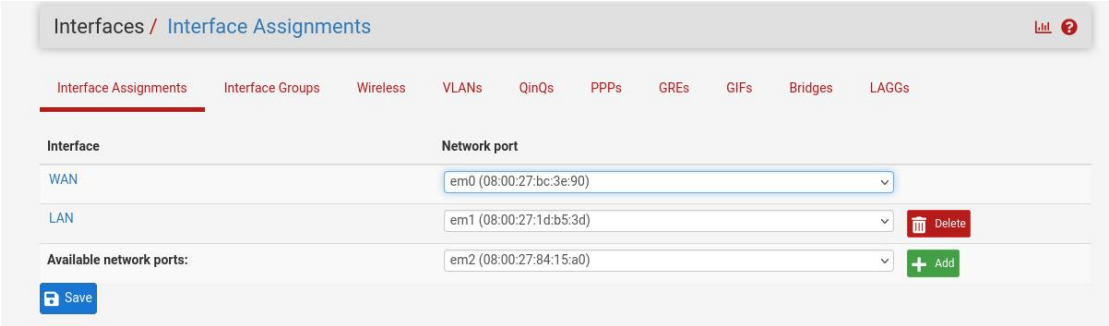
Ahora en nuestra kali vemos nuestra IP con ifconfig y es la 100.10
Vamos a desconectarnos y conectarnos de la red, para que nos asigne una nueva dirección IP del rango que hemos configurado en el servidor DHCP, que son la 100.10. hacemos ifocnfig para comprobar.

```
❯ ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:52:bd:ae:82  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.10  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::5d2e:960a:3176:4c0a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:b7:26:75  txqueuelen 1000  (Ethernet)
        RX packets 6971  bytes 3591505 (3.4 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 6574  bytes 743307 (725.8 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 69076  bytes 5955197 (5.6 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 69076  bytes 5955197 (5.6 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

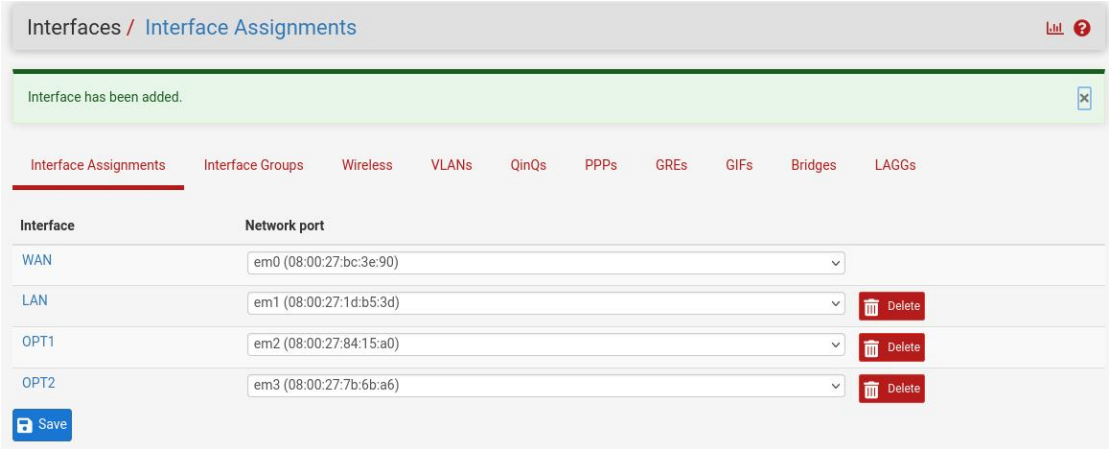
Ya tenemos definida nuestra red, ya tenemos el direccionamiento que queremos, ya hemos cumplido siguiente objetivo que nuestra maquina kali este en la red LAN.

Ahora volvemos a Pfsense para **configurar las DMZ y DMZ_2 de nuestra red.**

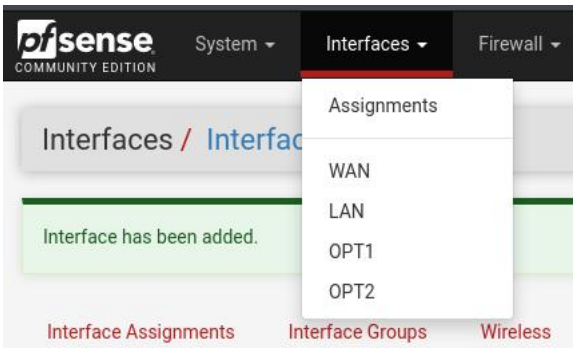
Interfaces > interfaces assignments (aquí vemos que podemos añadir mas interfaces de red)



Estas interfaces de red son las que podemos ver en la propia maquina de UTM (si vamos la configuración de la maquina, a red, donde definimos las 4 interfaces de red, ahora mismo solo estamos utilizando la 1 que seria la interfaz WAN y la 2 que seria la LAN) entonces vamos a añadir las que nos faltan, las DMZ, para ello le damos a añadir 2 veces y nos tienen que salir la em2 y la em3.

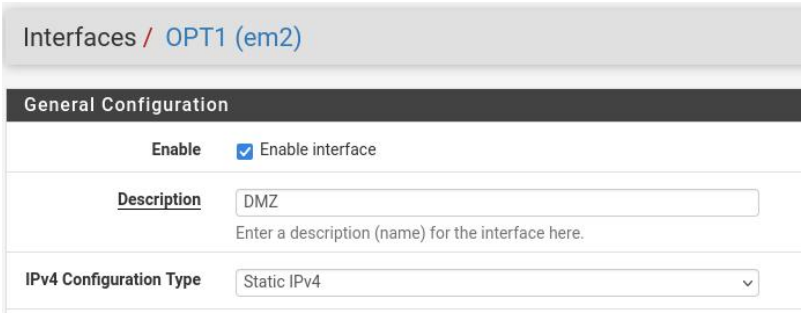


Vemos que ahora ya nos aparece OPT1 y OPT2, vamos a **configurar estas interfaces de red.**



Configuramos las interfaces → Interfaces-OPT1

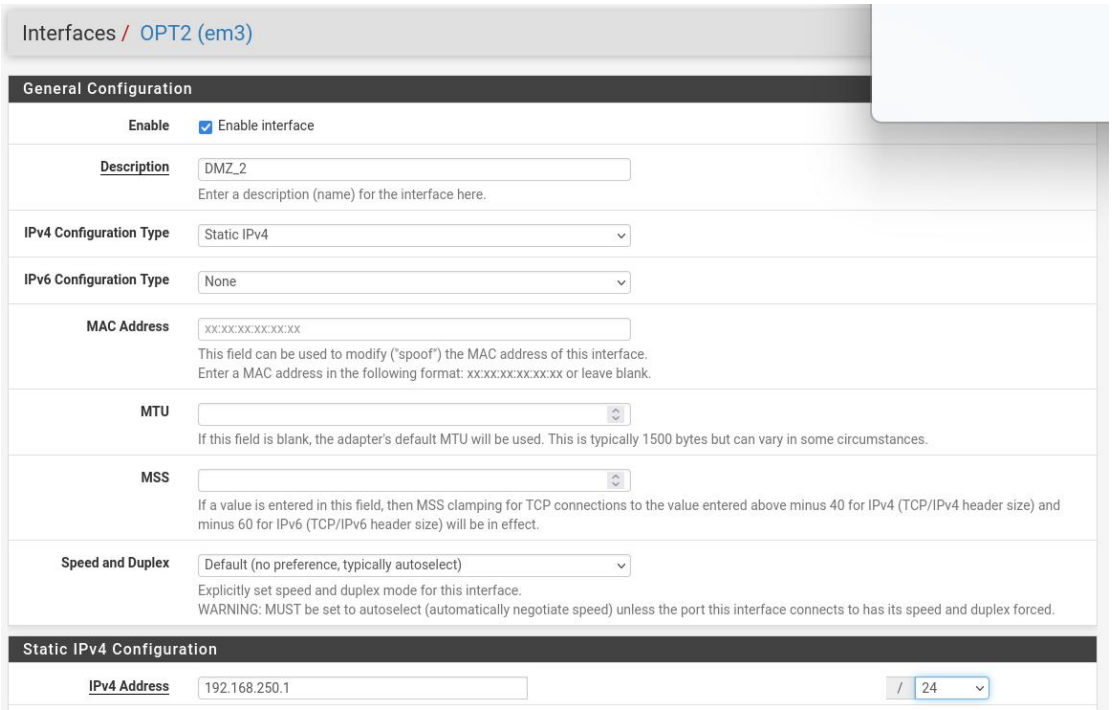
- **Enable** → Habilitamos la interface
- **Description** → DMZ
- **IPv4 Configuration type** → Static IPv4 (esta dirección es la que vemos en la maquina de la UTM, es la de la puerta de enlace, para que no cambie tiene que ser estática para las maquinas que hay en la red sea su dirección de salida del router)



- **IPv4 address** → 192.168.200.1/24

Configuramos las interfaces → Interfaces-OPT1

- Ahora vamos a OPT2, hacemos lo mismo pero cambiando el nombre a DMZ_2 y el rango 192.168.250.1/24



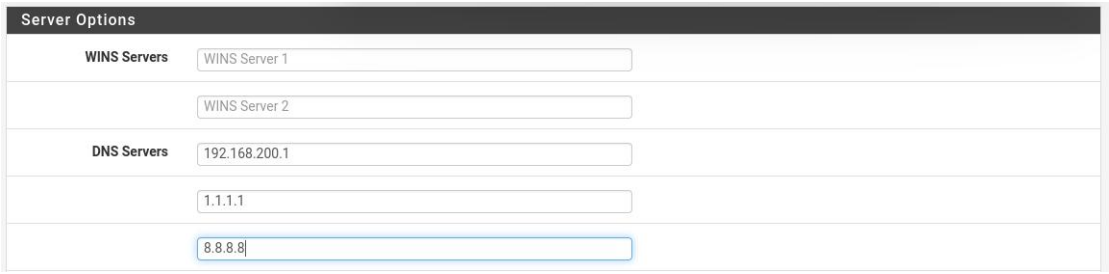
Si nos vamos a la maquina de UTM y le damos a intro ya nos aparecer nuestras 4 interfaces bien definidas con los rangos que queremos.

Vamos a la configuración del DHCP y habilitamos el DHCP server en esas interfaces

Vamos a configurar el direccionamiento que le vamos a dar a nuestros segmentos de red, que hemos definido en nuestro esquema, en nuestro DMZ y DMZ_2, junto con la habilitación del servidor DHCP.

- Enable **DHCP server** on DMZ interface
- **Range** 192.168.200.100 to 192.168.200.150

- **DNS servers**: 192.168.200.1 (la del pfsense), segundo servidor 1.1.1.1 y tercer servidor 8.8.8.8



- **Gateway** 192.168.200.1 (la del pfsense)

Ahora vamos a DMZ_2 y hacemos lo mismo

- El rango de direcciones IP 192.168.250.100 a 192.168.250.150

- En los servidores DNS 192.168.250.1. luego el 1.1.1.1 y el 8.8.8.8

- En la gateway 192.168.250.1

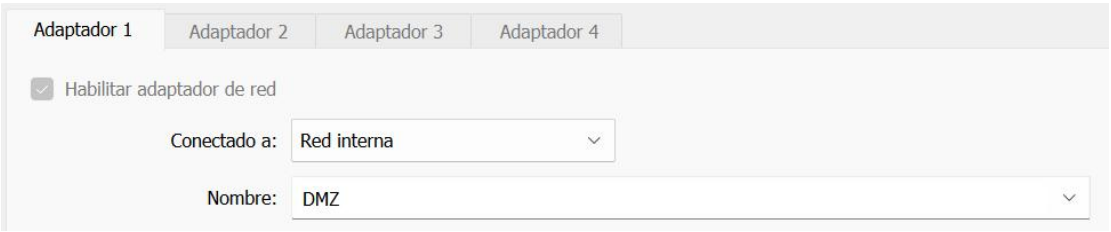
Ahora mismo tenemos configurado el servidor DHCP y el DNS.
Ya tenemos en los adaptadores LAN, DMZ y DMZ_2 configurados el rango de IP y el servidor DHCP.

Ya hemos asignado a las interfaces de red diferentes segmentos de IP, ahora vamos a comprobar si lo que hemos hecho funciona.

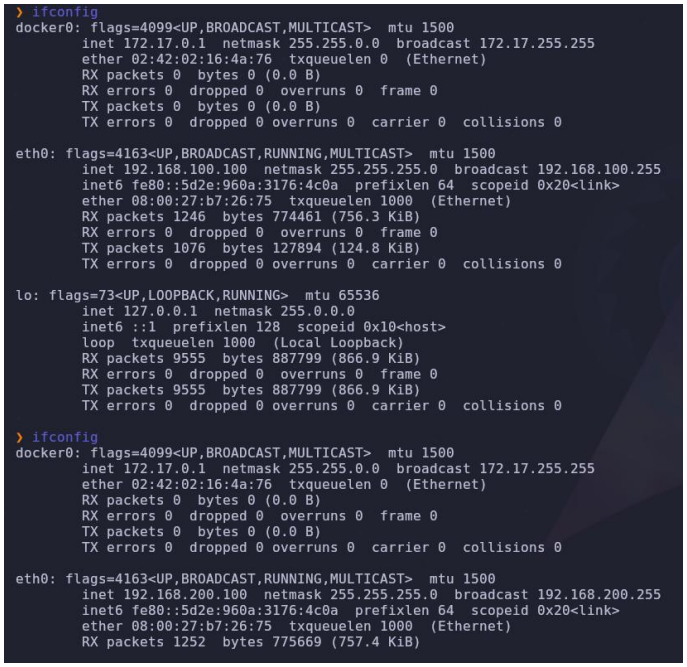
Para comprobar que si metemos algo en kali-2 (ver esquema), nos da un rango dinámico de direccionamiento de DHCP correspondiente a 192.168.200.100 a 192.168.200.150, le tendríamos que cambiar la puerta de entrada al kali que tenemos.

Si ahora cogemos la kali, y en vez de conectarnos a la LAN, la conectamos a la DMZ, el servidor DHCP nos debería dar automáticamente una nueva IP.

Para comprobar que funciona, nos vamos a la kali, configuración y hacemos un cambio de red (en el adaptador 1, donde está la LAN, ponemos DMZ).



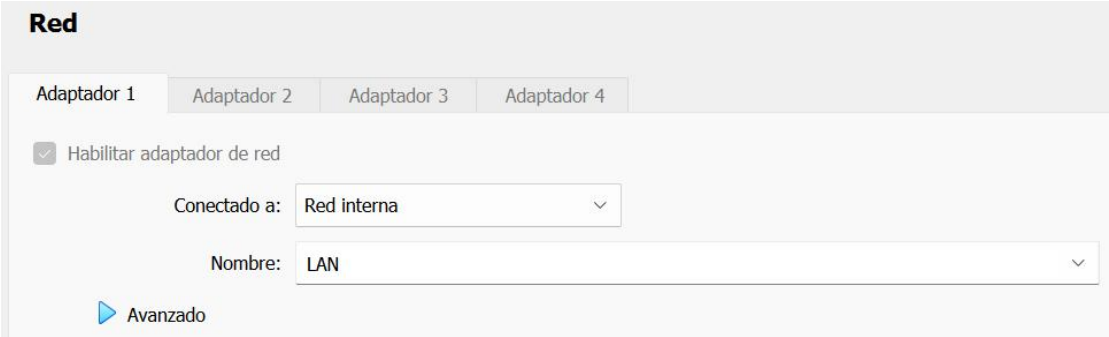
Si ahora hacemos un ifconfig deberíamos ver que nos ha cambiado la IP, y que la que no da está dentro del rango de las direcciones de DMZ que hemos definido.



Ya tenemos configuradas las redes y hemos comprobado que funciona.

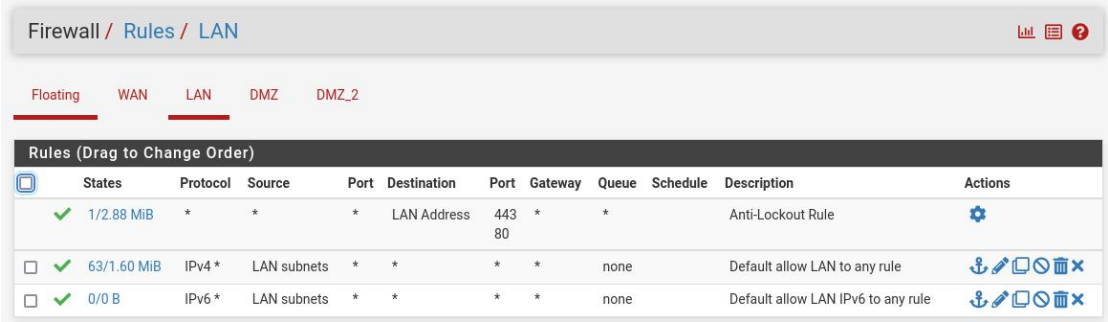
Con estas configuraciones vemos que no tenemos acceso a ping ni a internet.

Nos cambiamos a la red LAN para poder acceder todos bien al Pfsense y poder ver un poco lo que nos esta pasando ahora mismo.

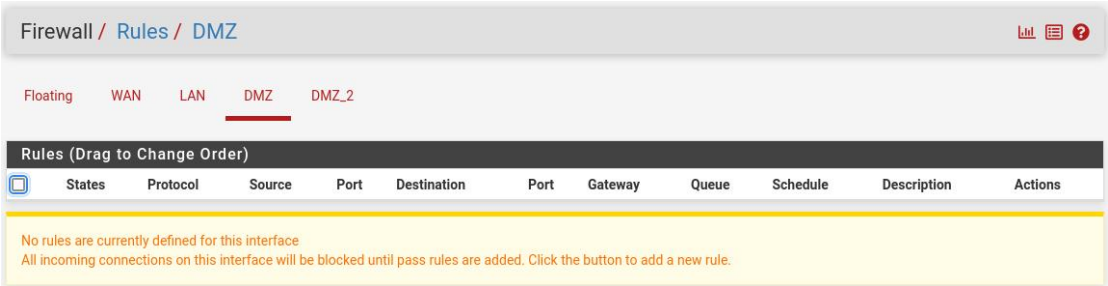


Creamos las **reglas de los FW** quedando la siguiente configuración:

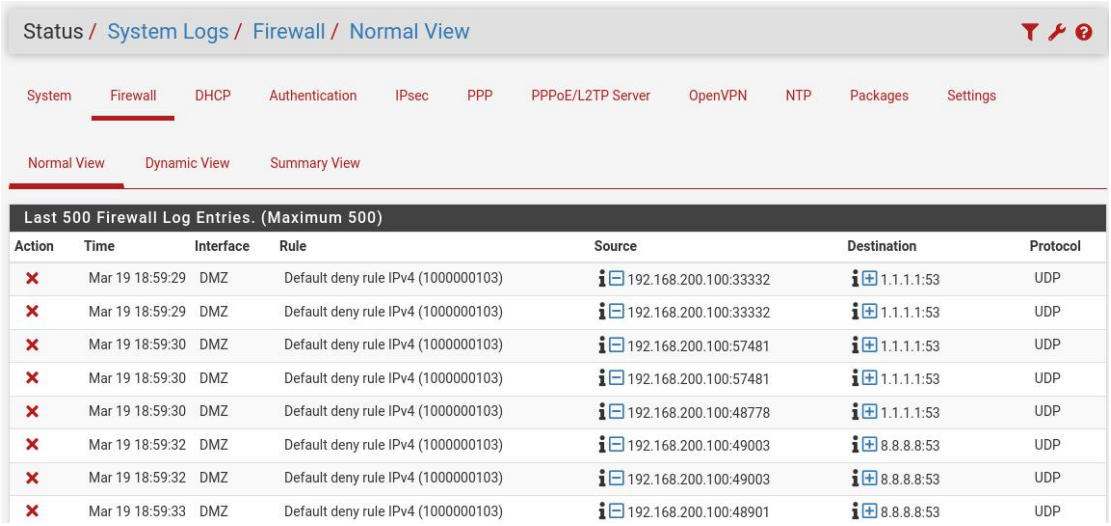
Una vez en Pfsense, si nos vamos a firewall-rules, vemos en la LAN por defecto tenemos reglas configuradas.



Pero en el resto de DMZ no tenemos nada.



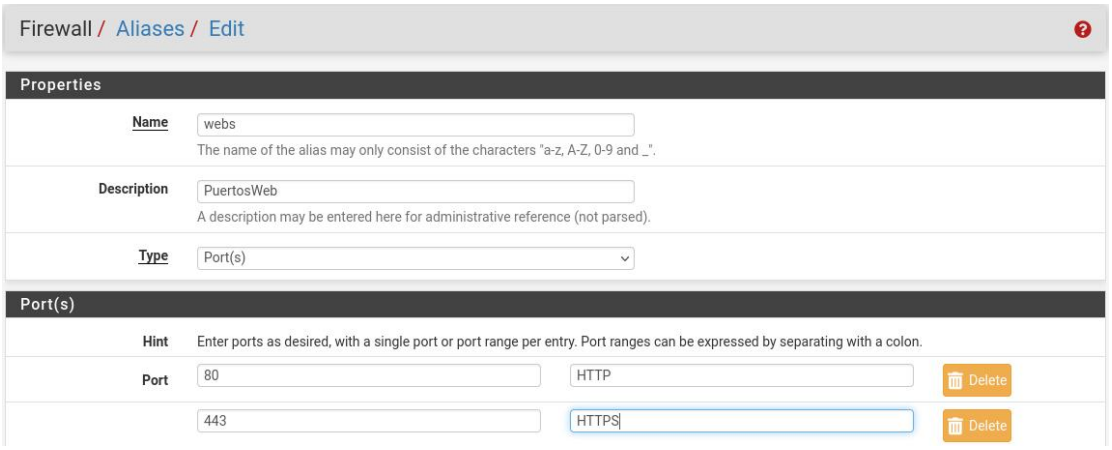
Los firewalls por defecto rechazan automáticamente todo el trafico, entonces al cambiarnos de red, si vamos a status-systems logs, y nos vamos a los logs de los firewalls vemos todas las peticiones DNS que hemos hecho están bloqueadas.



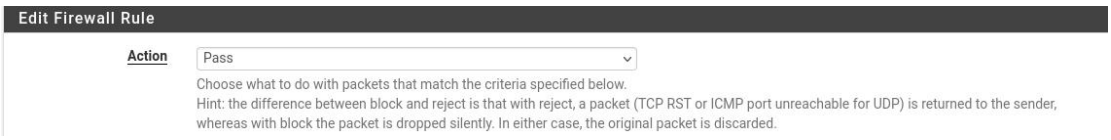
Configuración de los firewall:

Ahora vamos a hacer la configuración del firewall, para eso entramos en **firewall-rules**. y entramos a DMZ.
Para navegar a internet necesitamos habilitar el **protocolo** http, **puerto** 4430,80,53 de DNS para que se lleve a cabo la resolución de nombre.

Vamos a crear las reglas de los firewall, para ello vamos a entrar dentro de firewall-aliases, y en **puertos**. Aquí vamos a poner el puerto 80 y el 443.
Esto lo hacemos para cuando pongamos en una regla de firewall queramos habilitar el tráfico hacia la web, poniendo puertos web nos coge todo.
No ponemos el puerto DNS porque utiliza lo que se denomina UDP(conexión no orientada) y HTTP y HTTPS utiliza TCP (en TCP son las tramas lo que se trata es que no se pierda ningún tipo de paquete cuando se establece la conexión)

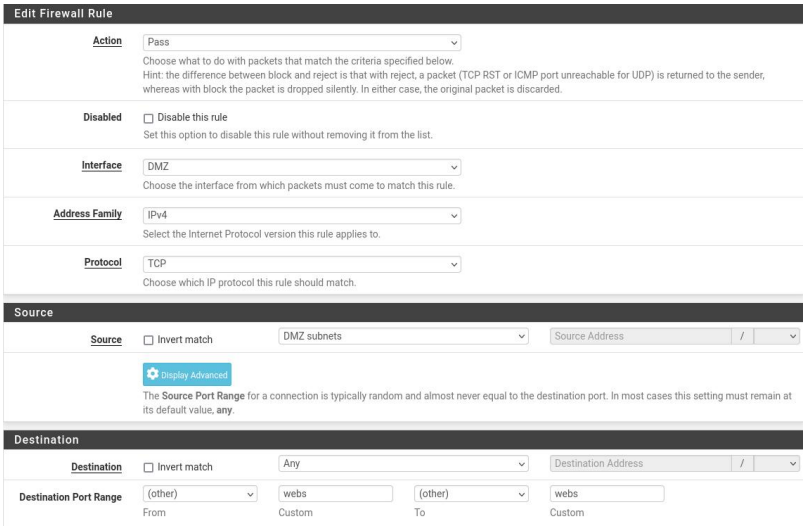


Ahora nos **vamos a crear las reglas en los firewalls**, nos metemos en firewall-rules. y nos metemos en DMZ, añadimos las reglas.
En **action** ponemos en pass(porque estamos creando una regla para permitir el trafico)

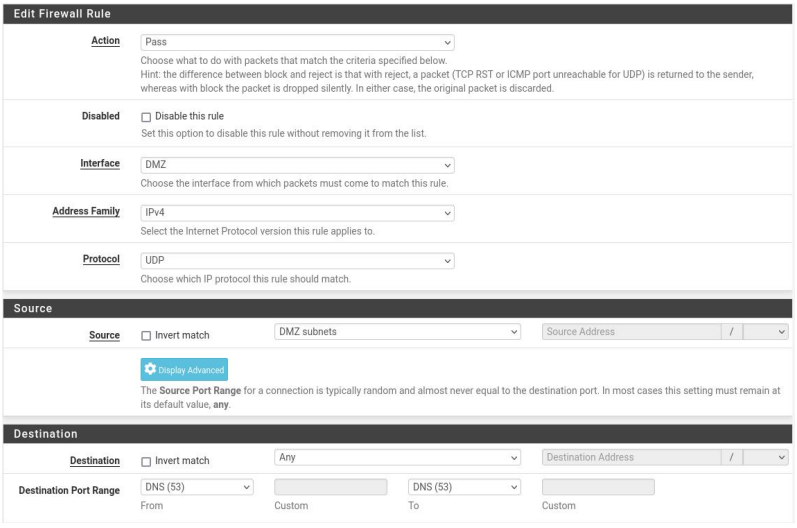


En **interface** ponemos DMZ, en **adress family** ponemos IPv4, en **protocol** ponemos TCP, para hacerlo mas seguro, en **source marcamos** DMZ_subnets, en **destination** ponemos from (others) en custom webs, y en to (other) y en custom webs.(hace referencia a los puertos que hemos definido antes en el alias puerto 443 y 80). en la **descripcion** podemos poner salida tráfico web.

Asi quedaría nuestra configuración:

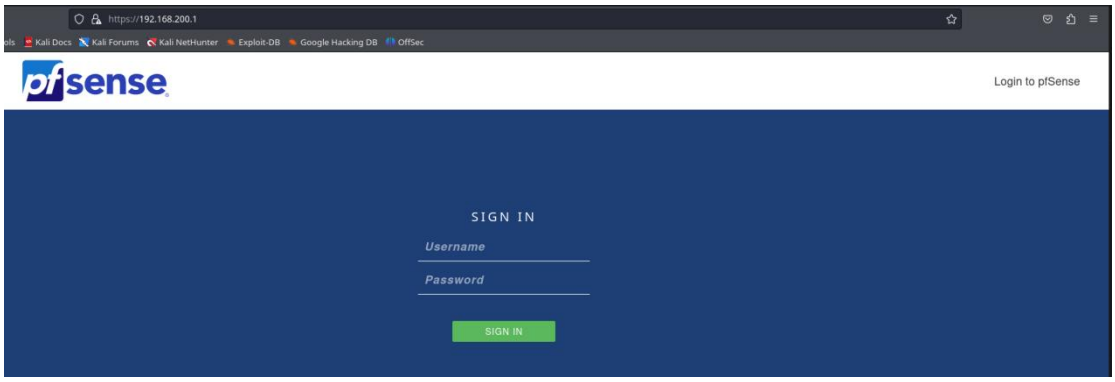


Ahora añadimos la regla que nos falta, que sería la del protocolo DNS, la de UDP. nos metemos en firewall-rules. y nos metemos en DMZ, añadimos las reglas.
Cambiamos el **protocolo** ponemos UDP en **destinations** ponemos from DNS(53) a to DNS(53) y en la **description** ponemos permitir trafico DNS.



Ahora vamos a ver si los cambios han funcionado, cambiamos la maquina de red a la DMZ para que nos asigne la nueva IP. ahora vamos a la consola y ponemos ip a. nos metemos en el navegador y vemos que funciona, ya tenemos acceso. y si queremos acceder al Pfsense (192.168.200.1) también tenemos acceso a la pagina del Pfsense.

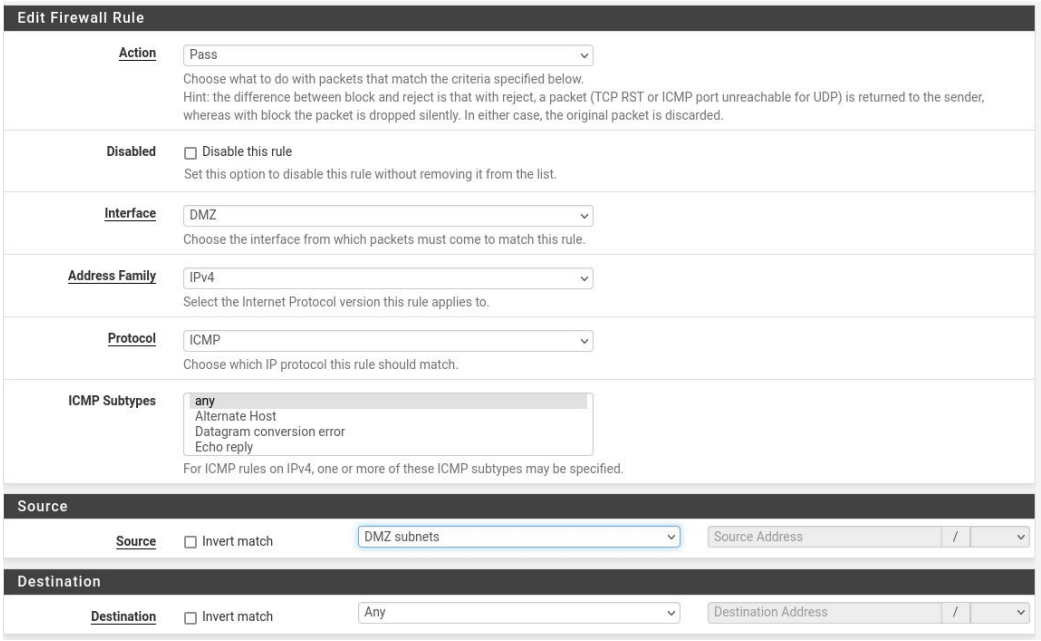
```
❯ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b7:26:75 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.100/24 brd 192.168.100.255 scope global dynamic noprefixroute eth0
       valid_lft 5781sec preferred_lft 5781sec
    inet6 fe80::5d2e:960a:3176:4c0a/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:02:16:4a:76 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
❯ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:b7:26:75 brd ff:ff:ff:ff:ff:ff
    inet 192.168.200.100/24 brd 192.168.200.255 scope global dynamic noprefixroute eth0
       valid_lft 7198sec preferred_lft 7198sec
    inet6 fe80::5d2e:960a:3176:4c0a/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:02:16:4a:76 brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
```



Ahora que estamos dentro de la red, vamos a ver en la consola y nos gustaría saber si los servidores de google están bien. hacemos ping 8.8.8.8 y vemos que no funciona, porque el ping no va ni por TCP, ni por UDP, va por lo que se llama ICMP.

```
❯ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
47 packets transmitted, 0 received, 100% packet loss, time 47096ms

~ x 1 48s
```

Para que nos funcione el ping deberíamos añadir otra **regla de permitido** en la DMZ desde el **protocolo** ICMP. En **source** ponemos desde la DMZ subnets y en **destino** lo dejamos en any.

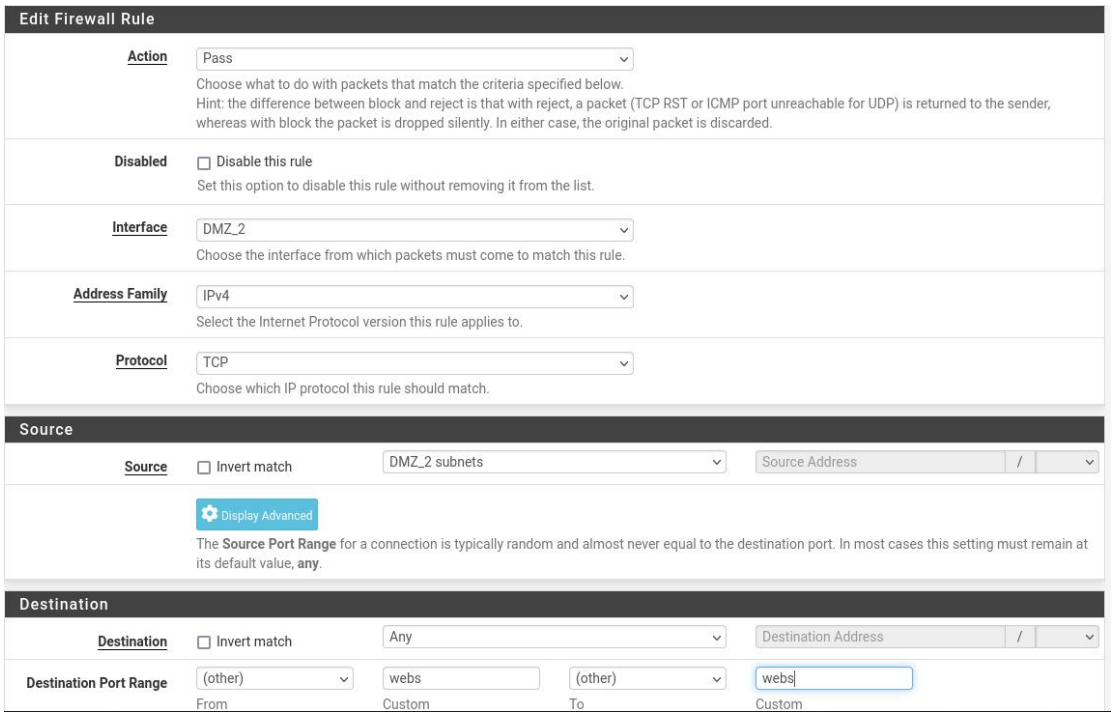


Si ahora hacemos ping ya nos responde.

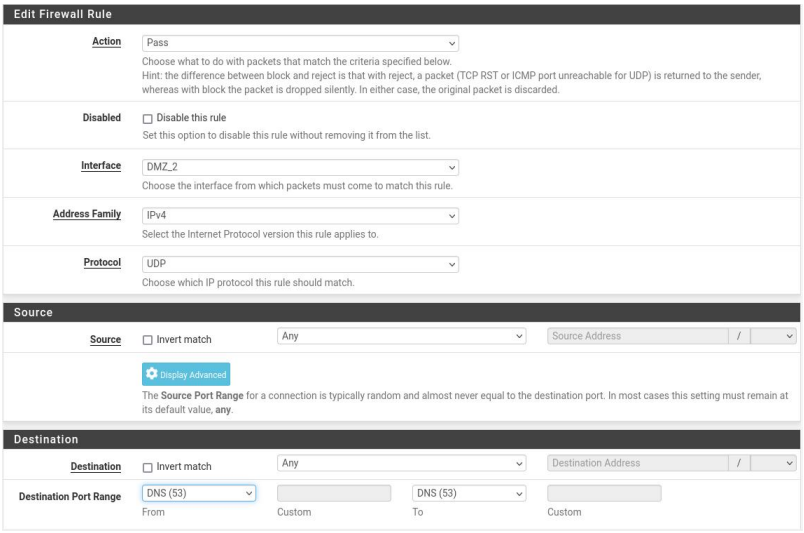
```
❯ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=115 time=11.9 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=115 time=10.3 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=115 time=8.16 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=115 time=8.37 ms
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 8.159/9.703/11.941/1.546 ms
```

Nos faltaría configurar la DMZ_2.

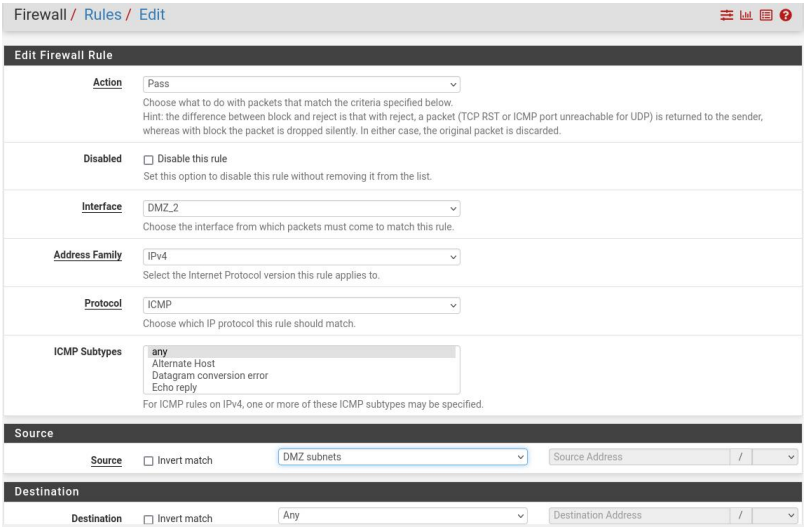
Para ello volvemos a las reglas de firewall y añadimos una nueva regla con el **protocolo**: TCP, **origen** DMZ_2 subnets, **destino**: any (destination port range: other, custom: webs, to: other, custom: webs).



Ahora añadimos una nueva regla con el **protocol**: udp, **origen**: any, **destino port range**: from dns(53) to dns (53).



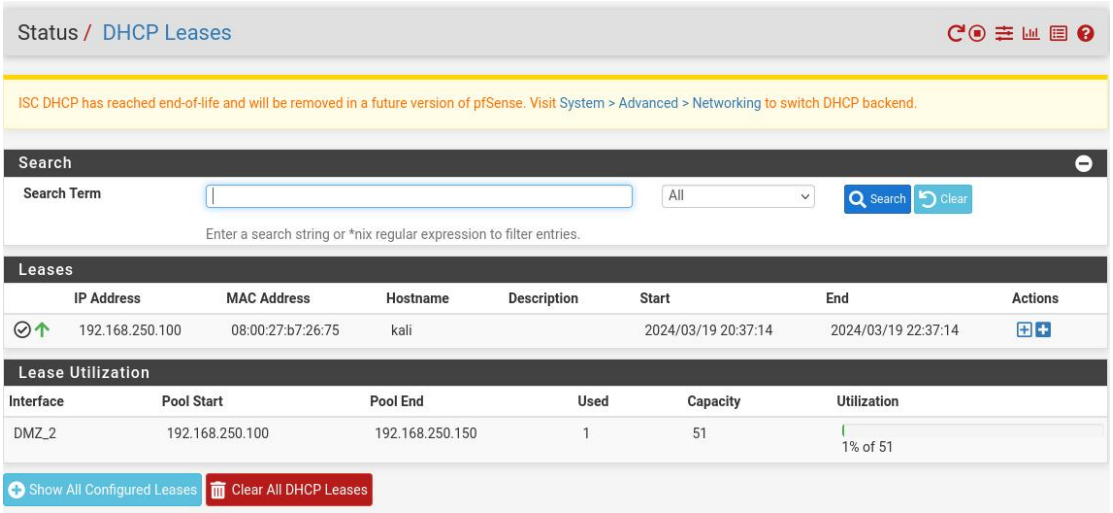
Y por ultimo añadimos otro para el protocolo ICMP, **protocolo**: ICMP, **origen**: DMZ_2 subnets y **destino**: any



El siguiente apartado que queremos conseguir es el kali

El apache es un servidor que vamos a tener dentro de nuestra red, así que nos interesa que siempre este dentro de la misma IP. si cambiamos constantemente el servdior de IP, al final no hay manera de encontrarlo, por lo que se suelen asignar IPs fijas a este tipo de maquinas.

Para ello nos vamos a conectar a la red DMZ_2, tenemos que ir al Pfsense, a status-dhcp leases, le damos a leave, y aquí vemos las maquinas que tenemos conectadas y con IP dentro de nuestro Pfsense, vemos la MAC de nuestra kali y la IP. esta IP esta asignada de manera dinámica por nuestro servidor de DHCP.



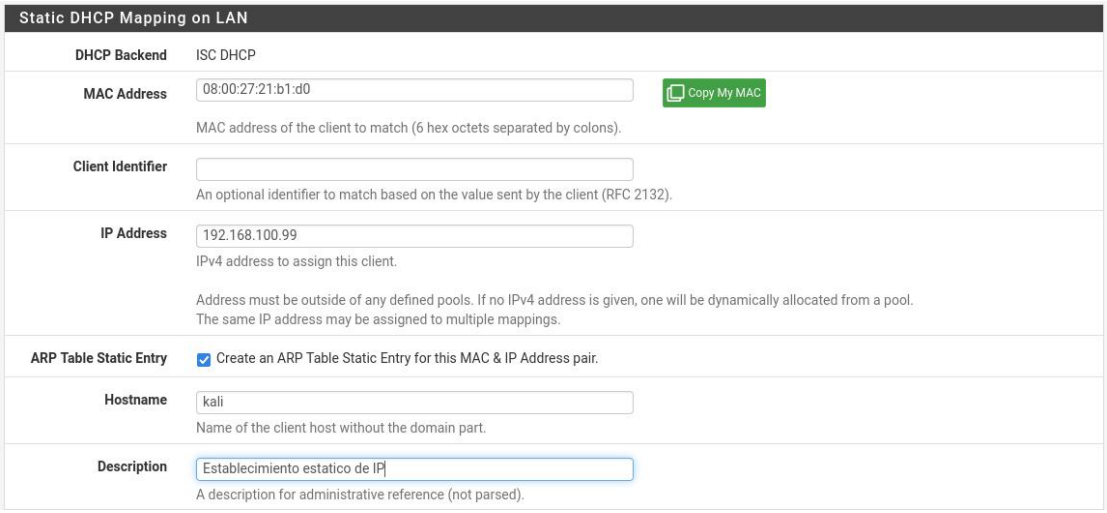
Puede ser útil para ademas de establecer direcciones IPs fijas, también establecer un match entre una dirección MAC y una IP para añadir una capa extra de seguridad.

Podemos, por ejemplo, configurar que para nuestra MAC, esta dirección IP es fija, es lo que se conoce como un filtro MAC.

Ahora vamos a añadir un mapeo estático de la dirección de nuestra kali, para que nuestro servidor apache no cambie.
Para ello le damos al + (el de fondo blanco) y aquí nos sale la MAC de nuestro servidor y la IP fija que queremos añadirle, en nuestro caso le añadimos la IP fija que pertenezca a nuestra subred.
lo que vamos a hacer es darle una IP que no entre en ese rango dinámico que vemos en la imagen.

Para que no entre en conflicto con el servidor DHCP, por ejemplo 192.168.250.99 (que esta fuera del rango de DHCP pero esta dentro del rango de nuestra red DMZ)

Le damos a **arp table static entry** (lo activamos, ruta estática de nuestra tabla de arp, para que nos asocie para siempre esa MAC a esa IP), podemos poner una **descripción** si queremos para tener una referencia establecimiento estático de IP.

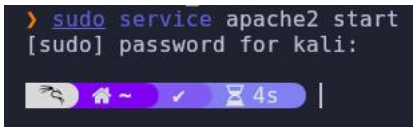


(ip Address: 192.168.250.99)

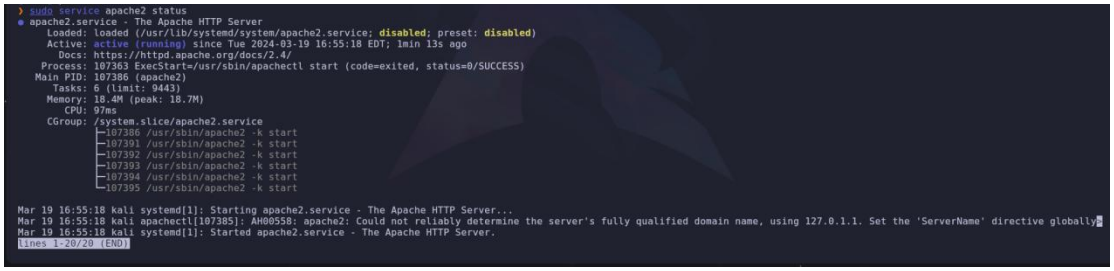
Desconectamos y conectamos y hacemos un ip a y vemos la 100.99 como dirección asignada en eth0.

APACHE:

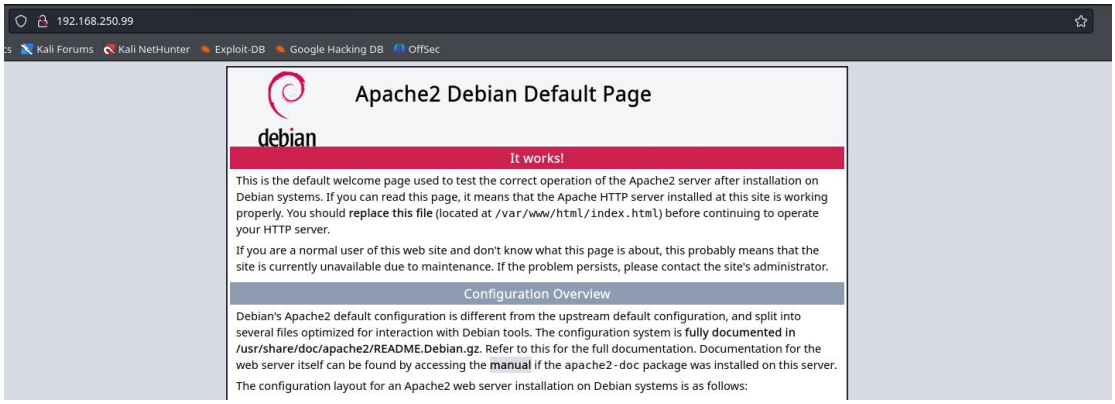
Ahora por último vamos a levantar nuesto servidor de apache, para ello hacemos un sudo service apache2 start



Si ahora hacemos un sudo service apache2 status deberíamos ver que esta corriendo.



Si en el navegador ponemos la IP 192.168.250.99 deberíamos ver que esta corriendo el servidor de Apache en la IP que le hemos asignado.



Tenemos una IP fija en la kali y servidor apache (lo tratamos como una única máquina, ver esquema), tenemos la kali asignada con una IP fija que es la 192.168.250.99, y si nos vamos a nuestra kali, poniendo esa ip deberíamos acceder a nuestro portal del Apache.

Vamos a hacer una configuración para que los que estén dentro de la DMZ_2 puedan acceder al servidor, y gente que esté detrás de nuestro UTM también puedan acceder.

Ahora vamos a conectarnos desde nuestro ordenador.
Tenemos que hacerle la petición a la WAN, del Pfsense, que es la que tiene contacto con el router de nuestra casa, la petición va a ir al router de nuestra casa y de ahí va a ir al Pfsense (la va a derivar).

Si ponemos la dirección de la WAN en el navegador, 192.168.1.55, no nos llega a la kali.

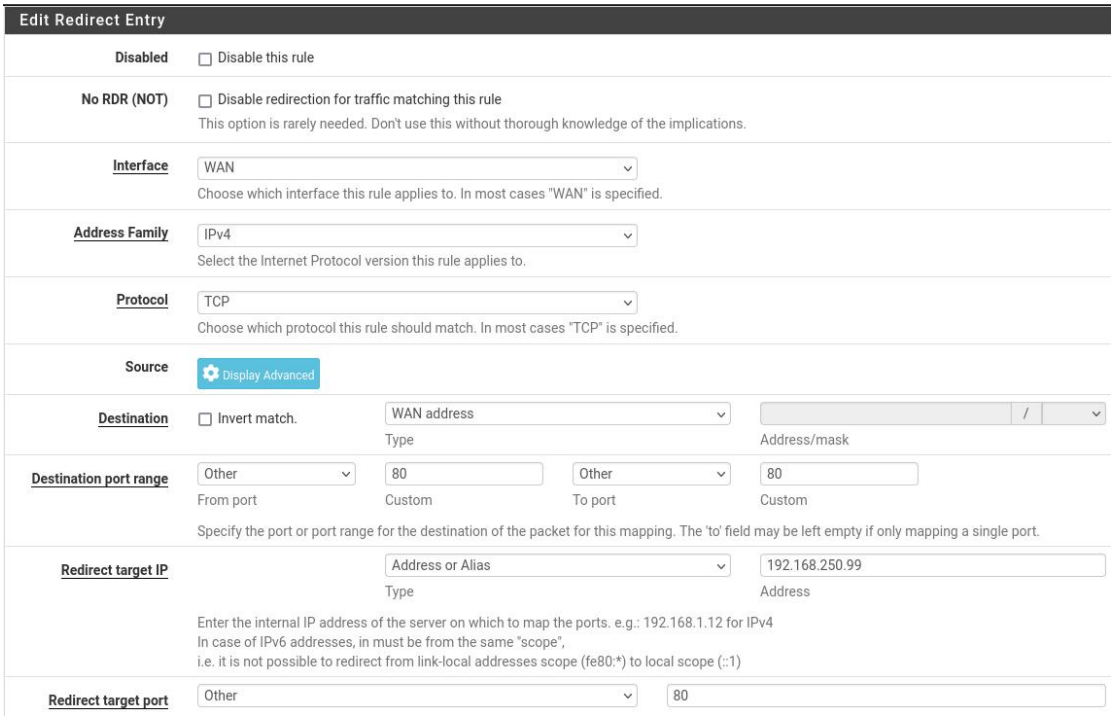
Esto es porque no hemos creado un túnel o un paso entre la WAN del Pfsense y la DMZ_2 del Pfsense.

Ahora mismo solo tenemos reglas de firewall para entrada y salida de trafico por el puerto 53, 80 y 443. como no estamos haciendo peticiones y estamos yendo solo a una única IP el Pfsense no es capaz de hacer nada con eso porque la IP a la que estamos dirigiéndonos es hacia la WAN del Pfsense.

Tenemos que crear una redirección para que el trafico que estamos enviando a la WAN del Pfsense nos lo encapsule y nos lo mande hacia la DMZ_2, lo redirija (lo que seria un nateo).

Lo que vamos a hacer en pfsense es crear una regla para decirle que todo el trafico que nos llegue al puerto 80 de nuestra interfaz WAN, nos lo reenvíe hacia nuestra DMZ_2. entonces vamos a ir a firewalls-Nat y añadimos una regla.

Aquí en **interface**: WAN, **protocolo**: TCP, **destino**: WAN address y **destination** port custom 80, custom 80, **redirección de IP**: address or alias 192.168.250.99 (nuestra kali), en **redirect target port**: other, custom: 80 (para crear la redirección), **description**: regla apache server

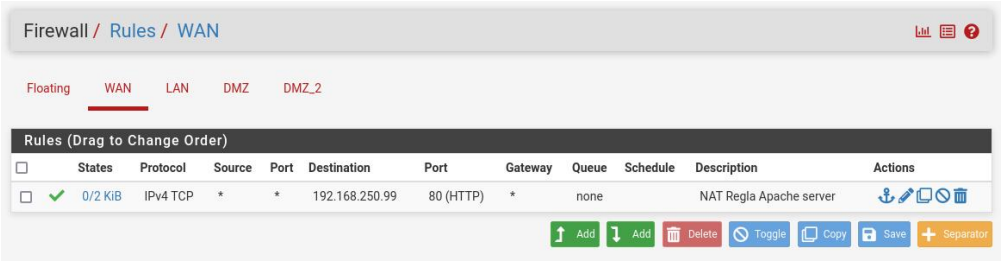


Lo que estamos diciendo es que el trafico que entre hacia la WAN con dirección de la WAN nos lo redireccione (nateo) hacia la IP de la kali que tenemos en nuestra DMZ_2 y al puerto web de la kali (puerto 80, donde esta nuestro servidor de apache).

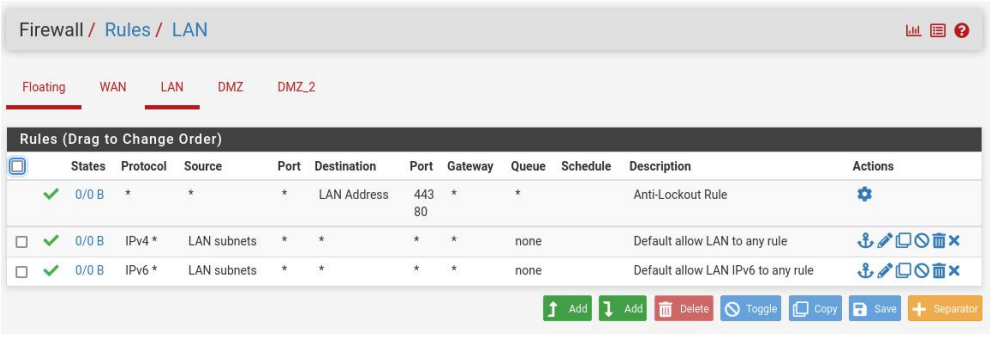
Si ahora nos vamos a nuestra maquina host y buscamos la IP de nuestra WAN (192.168.1.55), ya accede a nuestro Apache.

REGLAS DEL FIREWALL

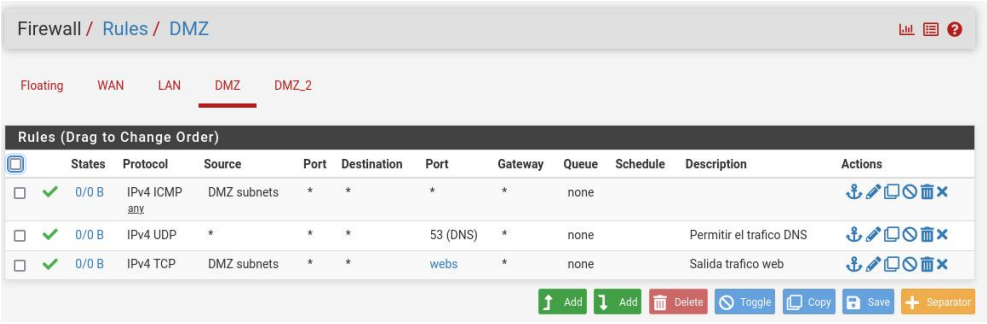
WAN:



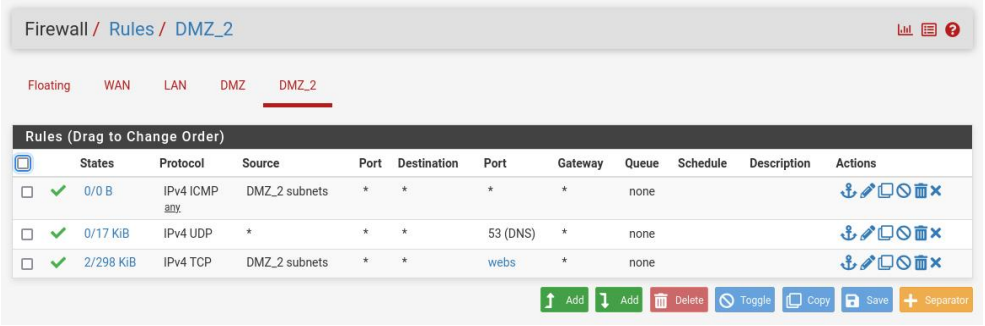
LAN:



DMZ:



DMZ_2:

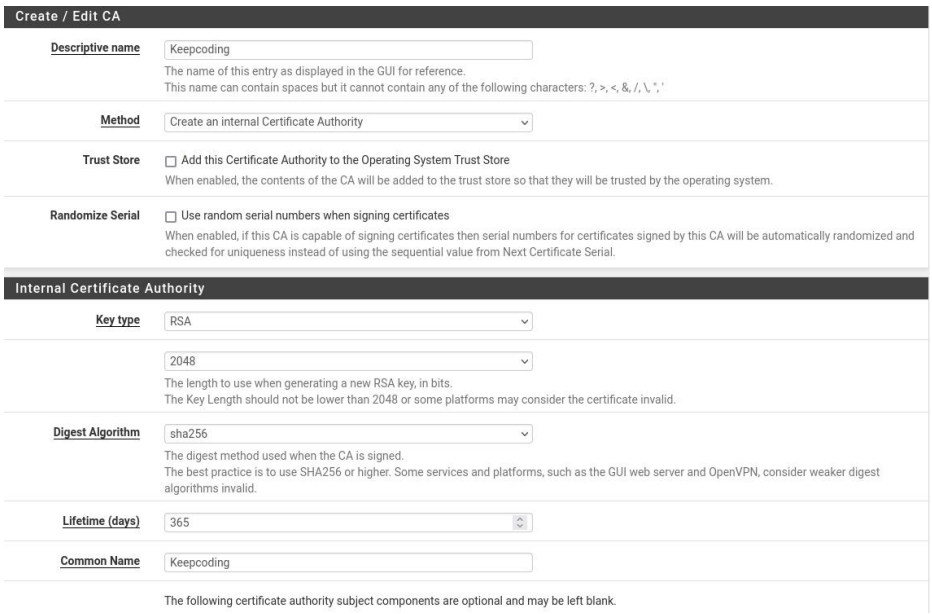


CREACION Y CONFIGURACION DE LA VPN:

Para ello tenemos que instalar un paquete que es el de openvpn, vamos en el pfsense a system-package manager-available packages y buscamos openvpn-client-export 1.9.2 (para poder exportar la configuración del cliente de vpn que vamos a configurar) y le damos a instalar.

Una vez instalado, vamos a ir a la instalación y creación de la vpn, para ello vamos a instalar los certificados con los que nos vamos a conecta a la vpn, cuando nos conectamos a la vpn necesitamos pasarle una serie de credenciales y de claves que confirmen que eres tú.

Vamos a generar los certificados para la conexion de la vpn, para ello vamos a system-certificates y tenemos que crear una entidad certificadora que es la que se va a encargar de generar y gestionar los certificados de nuestra vpn, le damos a add y en **descrptive name**: keepcoding, **method**: create an internal cert auth, **key type**: rsa 20148, **digest algorithm**: sha256, **lifetime**: 365, **common name**: keepcoding, **country code**: ES, **state or province**: Madrid, **city**: madrid, **organization**: keepcoding, **organization unit**: it

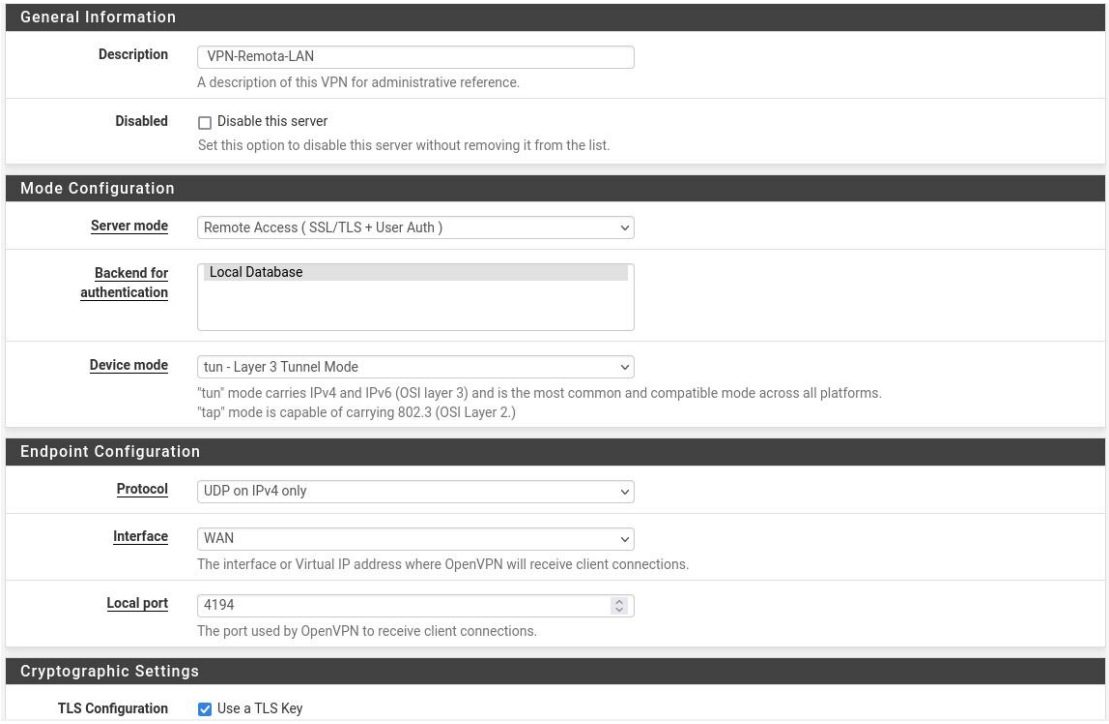


Ahora tenemos que crear el certificado, vamos a la pestaña de al lado, **certificates** y le damos add, **descriptive name**: vpn, **lifetime**: 365,

certificate authority: keepcoding (la que hemos creado antes), **common name**: vpn.keepcoding.local, **certificate type**: server certificate (IMPORTANTE porque este es el certificado que vamos a crear para nuestro propio servidor del pfsense y que luego vamos a exportar al exterior).

Ahora tenemos que crear el usuario y los rangos para la vpn que hemos definido, porque dentro de la vpn se va a crear un tunel, y en ese tunel tenemos que asignar una serie de direcciones IP, puerto que va a utilizar la vpn y toda la configuración.

Para ello vamos a ir **vpn-openvpn** y en servers le damos add, **description**: vpn-remota-lan, en **server mode**: remote access (ssl/tls + use auth), en **protocol**: tcp on ipv4 only, en **device mode**: tun - layer 3 tunnel mode (modo tunel, nivel 3 de la capa para conectar redes a redes y nivel 2 para tenerlo todo en el mismo fragmento de red), **protocol**: udp on ipv4 only, **intrface**: WAN, en **local port**(esto es lo que vamos a cambiar, es seguridad por oscuridad, viene por defecto el que todo el mundo usa para vpn, le cambiamos el puerto para ocultarla): 4194, en **server certificate**: vpn (server: yes, ca: keepcoding) (el que hemos creado antes), en **hardware crypto**: si nos sale el hardware para poder acelerar el proceso criptografico de la vpn lo seleccionamos, en **ipv4 tunnel network** (ponemos la direccion ip que va a utilizar nuestro tunel de la vpn): 192.168.220.0/24, en **ipv4 local network** (la red para que sea accesible desde el exterior ponemos la de la LAN: 192.168.100.0/24 (esto seria el punto accesible que va a tener nuestra vpn, toda la red que cuelga de la parte de la lan), en **redirect ipv4 gatewaty** (no lo activamos): si lo activamos se redirige todo el trafico de la red de los usuarios, si queremos tener controlado absolutamente todo lo que hagan los usuarios, lo redireccionas, en **concurrent conecctions** (gente que se puede conectar a la vez): 10



Automatically generate a TLS Key.

Peer Certificate Authority: Keepcoding

Peer Certificate Revocation list: No Certificate Revocation Lists defined. One may be created here: System > Cert. Manager

OCSP Check: Check client certificates with OCSP

Server certificate: VPN (Server: Yes, CA: Keepcoding)

Certificates known to be incompatible with use for OpenVPN are not included in this list, such as certificates using incompatible ECDSA curves or weak digest algorithms.

DH Parameter Length: 2048 bit

Diffie-Hellman (DH) parameter set used for key exchange.

ECDH Curve: Use Default

The Elliptic Curve to use for key exchange.
The curve from the server certificate is used by default when the server uses an ECDSA certificate. Otherwise, secp384r1 is used as a fallback.

Data Encryption Algorithms

Available Data Encryption Algorithms:
AES-128-CBC (128 bit key, 128 bit block)
AES-128-CFB (128 bit key, 128 bit block)
AES-128-CFB1 (128 bit key, 128 bit block)
AES-128-CFB8 (128 bit key, 128 bit block)
AES-128-GCM (128 bit key, 128 bit block)
AES-128-OFB (128 bit key, 128 bit block)
AES-192-CBC (192 bit key, 128 bit block)
AES-192-CFB (192 bit key, 128 bit block)
AES-192-CFB1 (192 bit key, 128 bit block)
AES-192-CFB8 (192 bit key, 128 bit block)

Click to add or remove an algorithm from the list

Allowed Data Encryption Algorithms. Click an algorithm name to remove it from the list:
AES-256-GCM
AES-128-GCM
CHACHA20-POLY1305

The order of the selected Data Encryption Algorithms is respected by OpenVPN. This list is ignored in Shared Key mode.

Fallback Data Encryption Algorithm: AES-256-CBC (256 bit key, 128 bit block)

The Fallback Data Encryption Algorithm used for data channel packets when communicating with clients that do not support data encryption algorithm negotiation (e.g. Shared Key). This algorithm is automatically included in the Data Encryption Algorithms list.

Auth digest algorithm: SHA256 (256-bit)

The algorithm used to authenticate data channel packets, and control channel packets if a TLS Key is present.
When an AEAD Encryption Algorithm mode is used, such as AES-GCM, this digest is used for the control channel only, not the data channel.
The server and all clients must have the same setting. While SHA1 is the default for OpenVPN, this algorithm is insecure.

Hardware Crypto: Intel RDRAND engine - RAND

Certificate Depth: One (Client+Server)

When a certificate-based client logs in, do not accept certificates below this depth. Useful for denying certificates made with intermediate CAs generated from the same CA as the server.

Strict User-CN Matching: Enforce match

When authenticating users, enforce a match between the common name of the client certificate and the username given at login.

Client Certificate Key Usage Validation: Enforce key usage

Verify that only hosts with a client certificate can connect (EKU: "TLS Web Client Authentication").

Tunnel Settings

IPv4 Tunnel Network: 192.168.220.0/24

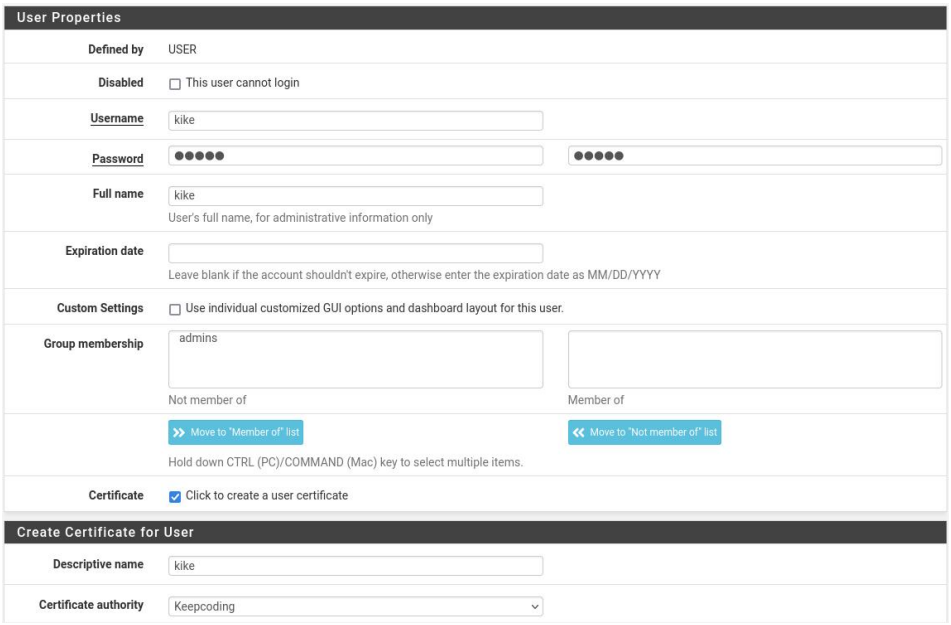
This is the IPv4 virtual network or network type alias with a single entry used for private communications between this server and client hosts expressed using CIDR notation (e.g. 10.0.8.0/24). The first usable address in the network will be assigned to the server virtual interface. The remaining usable addresses will be assigned to connecting clients.

A tunnel network of /30 or smaller puts OpenVPN into a special peer-to-peer mode which cannot push settings to clients. This mode is not compatible with several options, including Exit Notify, and Inactive.

IPv6 Tunnel Network

This is the IPv6 virtual network or network type alias with a single entry used for private communications between this server and client hosts

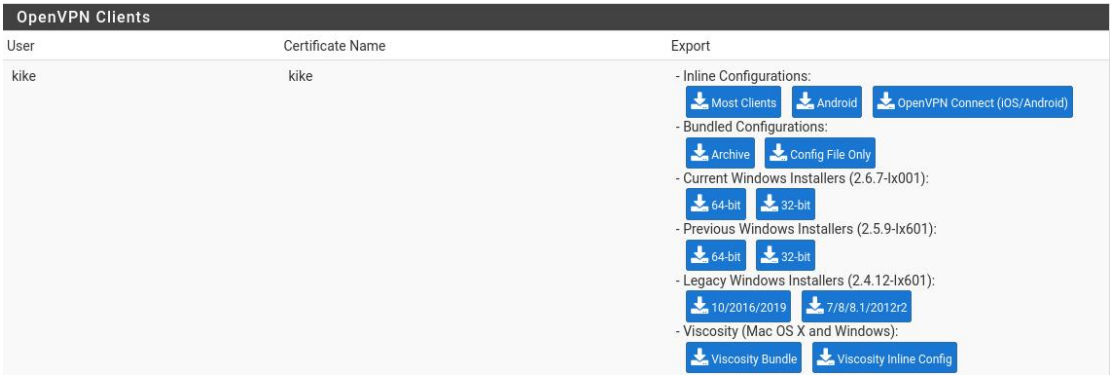
Ahora nos vamos a **system-user manager** y aqui añadimos un nuevo usuario que es nuestro usuario de la vpn.
le damos a add, **username**: kike, **password**: 12345 (nos tenemos que acordar), **full name**: kike, le damos a **certificate**: click to create a user certificate, se nos despliega un menu para crear el certificado del usuario para conectarnos con nuestra entidad de certificacion que hemos creado para generar estos certificados (maquinas a parte), **descriptive name**: kike, **certificate authority:** keepcoding, **key type**: rsa 2048, **digest algorithm** sha256, **lifetime**: 365, y le damos a guardar y ya tenemos nuestro usuario creado



Ahora lo que nos queda seria exportar el certificado de nuestro usuario, para ello nos vamos a **vpn-openvpn** y en la barra roja de arriba a **clien expor**t.

Seleccionamos la vpn que hemos creado con el puerto que hemos configurado en **remote access serve:** vpn-remota-lan upd4:4194.

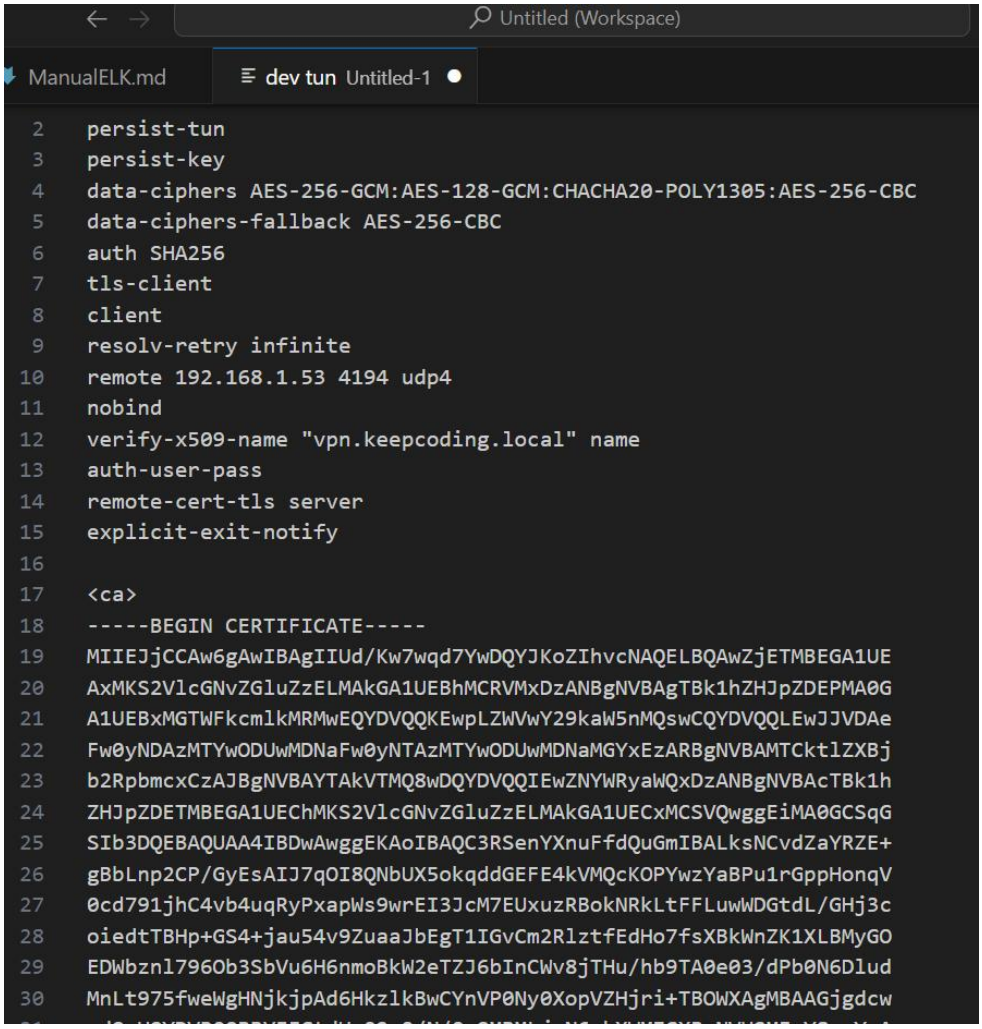
En openvpn clients nos tendría que salir el usuario que hemos creado y las configuraciones que hemos creado.



Descargamos **most clients**. se nos descarga un archivo **.ovpn**. ya tenemos el certificado descargado, lo abrimos con el editor de texto, y nos sale en la consola el certificado.

```
dev tun
persist-tun
persist-key
data-ciphers AES-256-GCM:AES-128-GCM:CHACHA20-POLY1305:AES-256-CBC
data-ciphers-fallback AES-256-CBC
auth SHA256
tls-client
client
resolv-retry infinite
remote 192.168.1.53 4194 udp4
nobind
verify-x509-name "vpn.keepcoding.local" name
auth-user-pass
remote-cert-tls server
explicit-exit-notify

<ca>
-----BEGIN CERTIFICATE-----
MIIEJjCCAw6gAwIBAgIIUd/Kw7wqd7YwDQYJKoZIhvcNAQELBQAwZjETMBEGA1UE
AxMKS2VlcGNvZGluZzELMAkGA1UEBhMCRVMxDzANBgNVBAgTBk1hZHJpZDEPMA0G
A1UEBxMGTWFkcmlkMRMwEQYDVQQKEwpLZWVwY29kaW5nMQswCQYDVQQLEwJJVDAe
Fw0yNDAzMTYwODUwMDNaFw0yNTAzMTYwODUwMDNaMGYxEzARBgNVBAMTCktlZXBj
b2RpbmcxCzAJBgNVBAYTAkVTMQ8wDQYDVQQIEwZNYWRyaWQxDzANBgNVBAcTBk1h
ZHJpZDETMBEGA1UEChMKS2VlcGNvZGluZzELMAkGA1UECxMCSVQwggEiMA0GCSqG
SIb3DQEBAQUAA4IBDwAwggEKAoIBAQC3RSenYXnuFfdQuGmIBALksNCvdZaYRZE+
gBbLnp2CP/GyEsAIJ7qOI8QNbUX5okqddGEFE4kVMQcKOPYwzYaBPu1rGppHonqV
0cd791jhC4vb4uqRyPxapWs9wrEI3JcM7EUxuzRBokNRkLtFFLuwWDGtdL/GHj3c
oiedtTBHp+GS4+jau54v9ZuaaJbEgT1IGvCm2RlztfEdHo7fsXBkWnZK1XLBMyGO
EDWbznl796Ob3SbVu6H6nmoBkW2eTZJ6bInCWv8jTHu/hb9TA0e03/dPb0N6Dlud
MnLt975fweWgHNjkjpAd6HkzlkBwCYnVP0Ny0XopVZHjri+TBOWXAgMBAAGjgdcw
gdQwHQYDVR0OBBYEFGtdHp02×9/N/0eSMBMLjsN6qkXWoWqkaDBmMRMwEQYDVQQDEwpLZWVwY29kaW5n
FGtdHp02×9/N/0eSMBMLjsN6qkXWoWqkaDBmMRMwEQYDVQQDEwpLZWVwY29kaW5n
MQswCQYDVQQGEwJFUzEPMA0GA1UECBMGTWFkcmlkMQ8wDQYDVQQHEwZNYWRyaWQx
EzARBgNVBAoTCktlZXBjb2RpbmcxCzAJBgNVBAsTAklUggghR3rDvCp3tjAMBgNV
HRMEBTADAQH/MAsGA1UdDwQEAwIBBjANBgkqhkiG9w0BAQsFAAOCAQEALbjWnRvP
7MjblFOTTXIBCLiIx3fIMTafDpBUyViZyta8kfAQy9NVVqr31YE9yjDVjQkEyKdh
KE9y7JvYdUY9PlyDoCxL2P5GFxHA16GiLl9f8ipNNe20hE31lWVVGKGhhLYVsDRV
AetYG5kxPcZZOXLR120eCNPSzcJmjiaS6Lv2kZxXYL5TYJrY1N+RHXAckxi+8VWZ
PmxXZN1owlY77UJTeAMigiJ8iWGmyOkVBBoIVJk3nFPq6gdPC63bH5i4SygN2zjH
h9EXGOx/g+s5pDwJbo59qBACjP88yg1vWKv4eu970I7vNgKqI0gClg15VzR2e19c
ze3Kc+kxlctyQg==
-----END CERTIFICATE-----
```

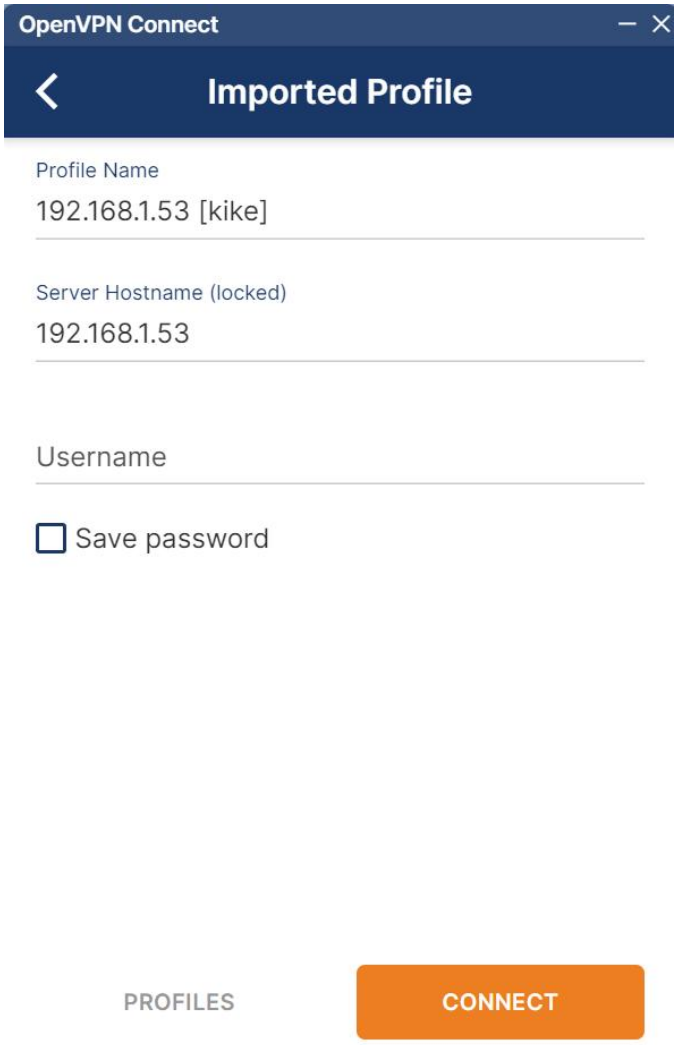
Copiamos el contenido y lo tenemos que tener en nuestra maquina host para conectarnos desde ella. Abrimos un editor, por ejemplo el Visual Studio Code y lo pegamos.



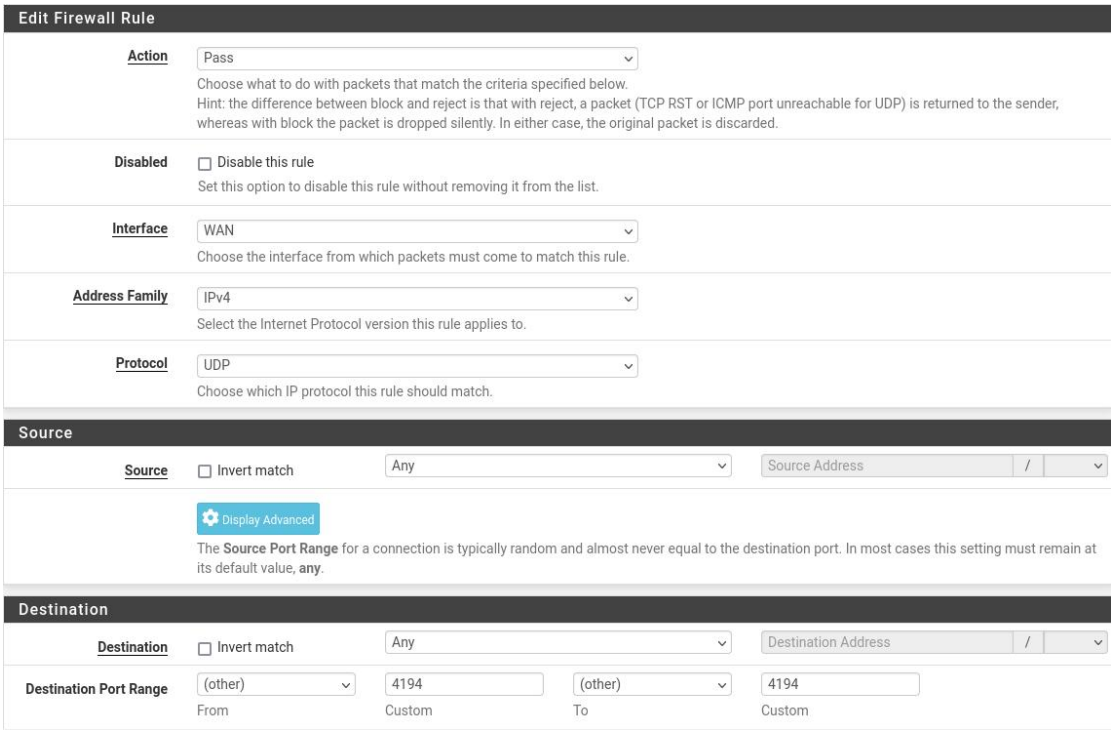
ahora le damos a file-save as (lo guardamos en un sitio donde lo veamos, con formato no extension (*,)y con el nombre por ejemplo kike y la extension ovpn: kike.ovpn)

Ya con el certificado en el escritorio de windows, nos tenemos que descargar la aplicación de **vpnclient** y nos bajamos el cliente en windows (en el host en el que estemos) (este es el cliente de vpn al que le vamos a subir el certificado de nuestra vpn).

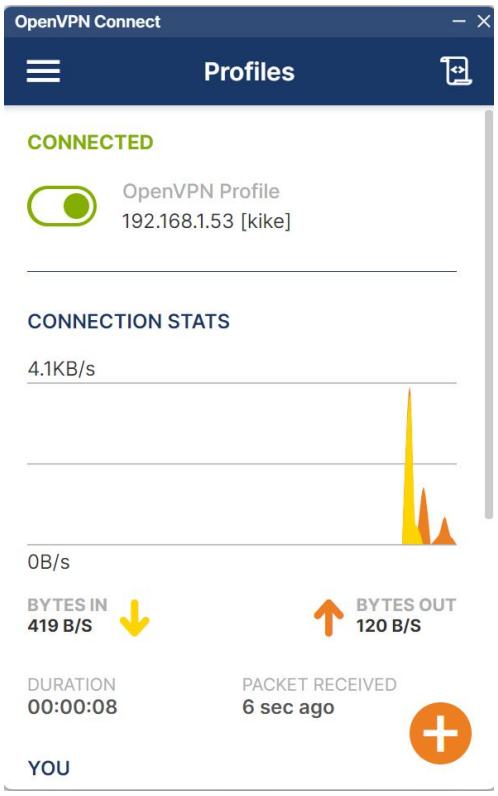
Abrimos la aplicación de el cliente y en upload file arrastramos el archivo con el certificado que hemos exportado, y nos debería salir la dirección de la WAN 192.168.1.53 y en el **nombre de usuario**(el que hemos configurado): kike, y le damos a connect y nos pide la **contraseña**: 12345, no se nos conecta porque nos falta la regla de firewall para habilitar el trafico de la vpn.



Vamos al pfsense firewall-rules y en la wan (barra de arriba) añdimos una nueva regla de firewall. **action**: pass, **interface**: wan, **address family**: ipv4, **protocol**: UDP, **source**(origen): any, **destination**: custom 4194 custom 4194(puerto que hemos configurado en nuestra vpn), **description**: regla WAN vpn.



Si volvemos a vpn ya se nos conecta.



HONEYPOT-COWRIE:

El honeypot que vamos a lanzar en dmz es el cowrie “ssh” a través el docker. Vamos a mandar los logs generados al archivo **cowrie.log**.

```
❯ sudo docker run -p 222:2222 cowrie/cowrie > cowrie.log
[sudo] password for kali:
/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:106: CryptographyDe
precationWarning: Blowfish has been deprecated
  b"blowfish-cbc": (algorithms.Blowfish, 16, modes.CBC),
/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:110: CryptographyDe
precationWarning: CAST5 has been deprecated
  b"cast128-cbc": (algorithms.CAST5, 16, modes.CBC),
/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:115: CryptographyDe
precationWarning: Blowfish has been deprecated
  b"blowfish-ctr": (algorithms.Blowfish, 16, modes.CTR),
/cowrie/cowrie-env/lib/python3.11/site-packages/twisted/conch/ssh/transport.py:116: CryptographyDe
precationWarning: CAST5 has been deprecated
  b"cast128-ctr": (algorithms.CAST5, 16, modes.CTR),
```

El primer paso que hemos hecho es lanzar el contenedor docker para que arranque con este comando: docker run -p 222:2222 cowrie/cowrie. el puerto de la izquierda es del anfitrión y el de la derecha del invitado.

Con docker ps vemos que tenemos el contenedor corriendo.

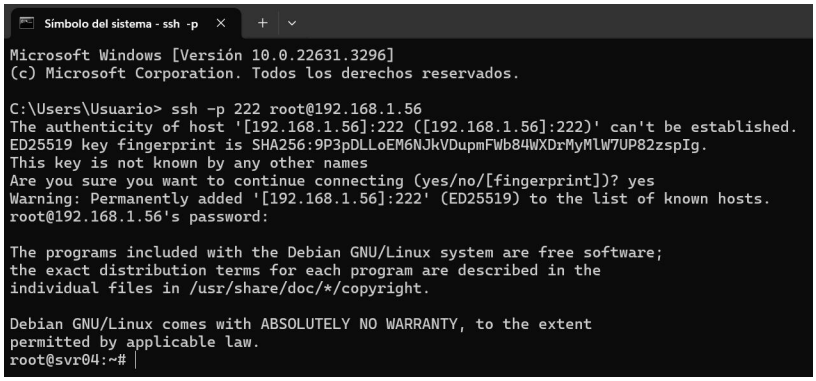
```
❯ sudo docker ps
[sudo] password for kali:
CONTAINER ID   IMAGE          COMMAND                  CREATED         STATUS         PORTS                                                  NAMES
3d3890f848bc   cowrie/cowrie  "/cowrie/cowrie-env/…"   3 minutes ago   Up 3 minutes   2223/tcp, 0.0.0.0:222->2222/tcp, :::222->2222/tcp   stoic_margulis
```

El que tenemos corriendo es el que hemos puesto que simula un honeypot de ssh y lo que le estamos diciendo es que nos lo abra en el puerto 222

Ahora lo que vamos a hacer es a través del cmd de nuestra maquina (windows), tenemos que hacer un ssh al puerto 222 con el usuario root y con la ip de nuestra maquina.

```
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.56  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::5d2e:960a:3176:4c0a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:b7:26:75  txqueuelen 1000  (Ethernet)
        RX packets 20653  bytes 6035850 (5.7 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20643  bytes 2512761 (2.3 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Abrimos cmd y ponemos ssh -p 222 root@192.168.1.56 (para conectarnos desde la maquina windows) y nos debería salir la clave para poder establecer la sesión. Nos dice que el fingerprint no se conoce, es nuevo, y por lo tanto nos pregunta si estamos seguros de conectarnos, le damos a si y en la contraseña podemos poner lo que sea y ya estamos dentro.



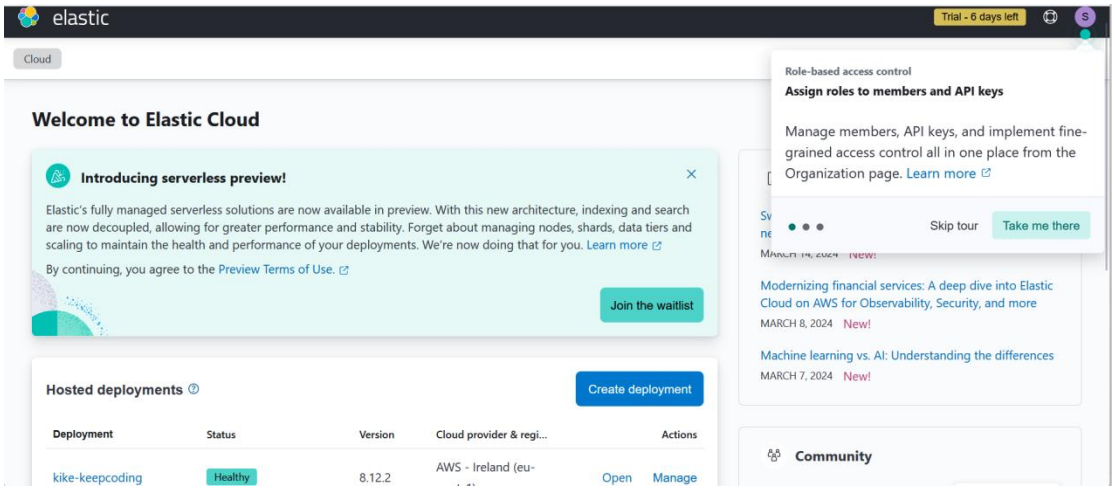
Este honeypot consiste en capturar todos los comandos que se prueban dentro del servidor, si hacemos un ls en el cmd de windows nos sale en la consola del kali conectada.

Es como simular un sistema de un servidor ssh y por lo tanto podríamos capturar los comandos, herramientas que intenten utilizar en nuestro sistema vulnerable y utilizar esa información para hacer threat hunting.

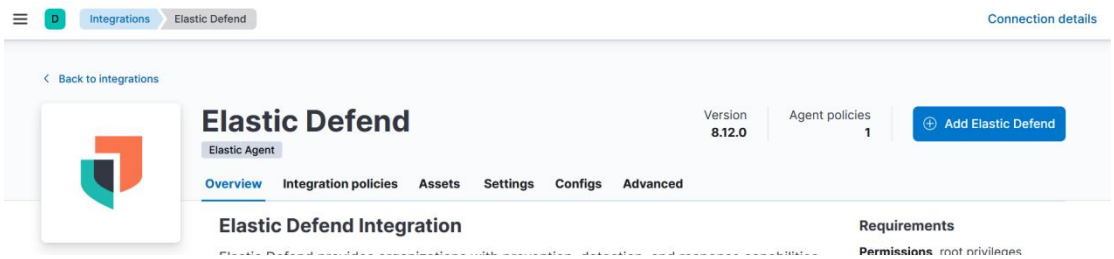
ELASTIC (SIEM):

La idea es que sepamos poder integrar fuentes, y mirar los logs que se recopilan.

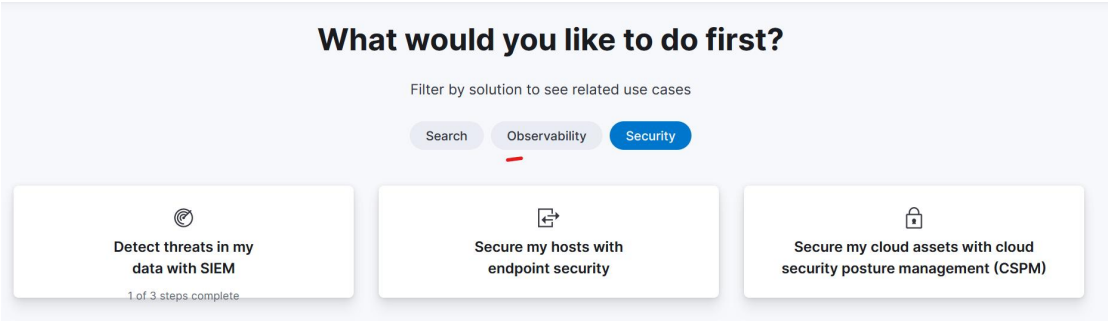
Entramos en elastic.



Vamos a añadir **Elastic Defend**.
Después deberemos copiar los comandos que nos dan según el sistema operativo en el que lo vayamos a instalar, en nuestro caso lo haremos en windows y linux.



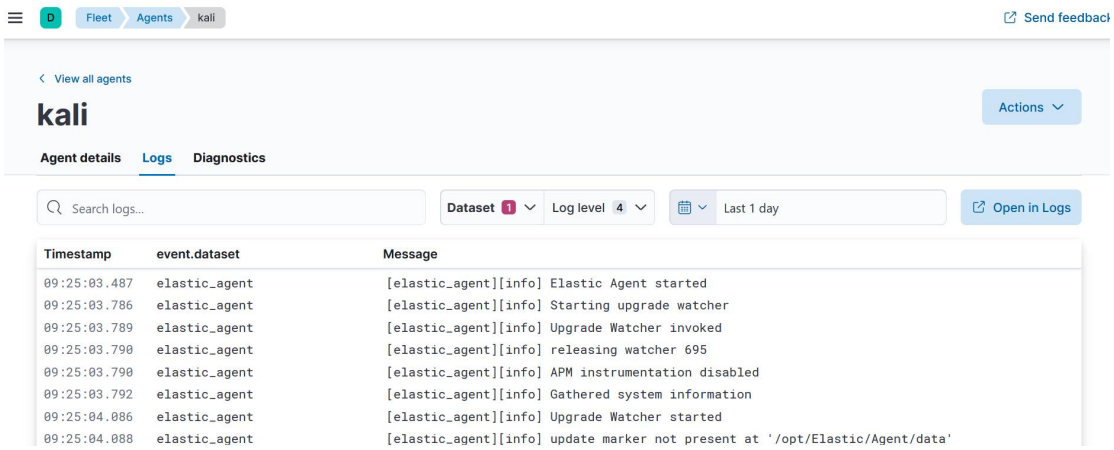
Aquí vamos a **detect threats in my data with siem**.



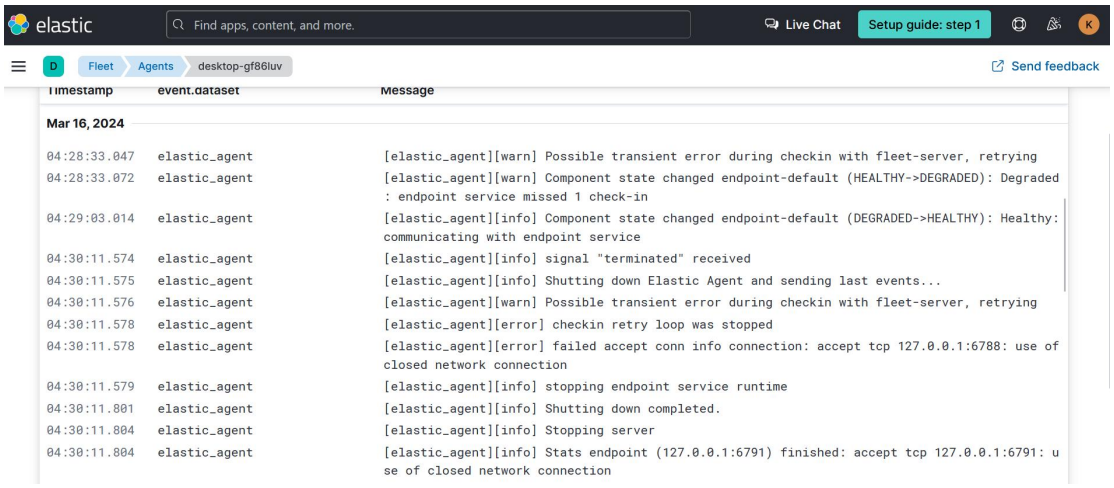
Ahora le damos a **add integration**, aquí ya lo que estamos haciendo es recopilar todos los eventos tanto de linux como de windows para recolectarlos en el siem.

Le estamos diciendo que coja todos los logs importantes o necesarios para el tema de seguridad.

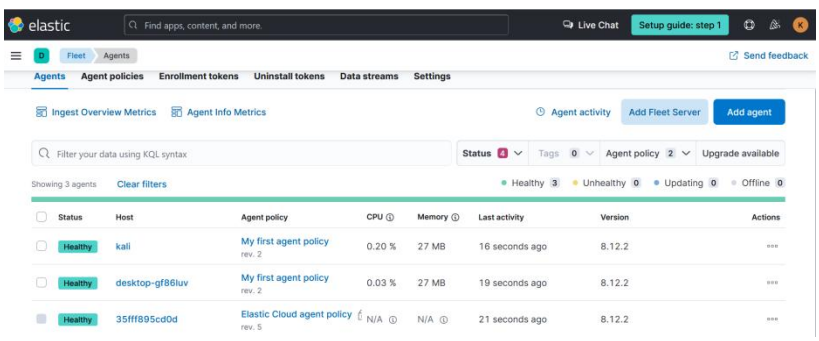
Aquí podemos ver los eventos de kali:



Aquí podemos ver los logs de windows:

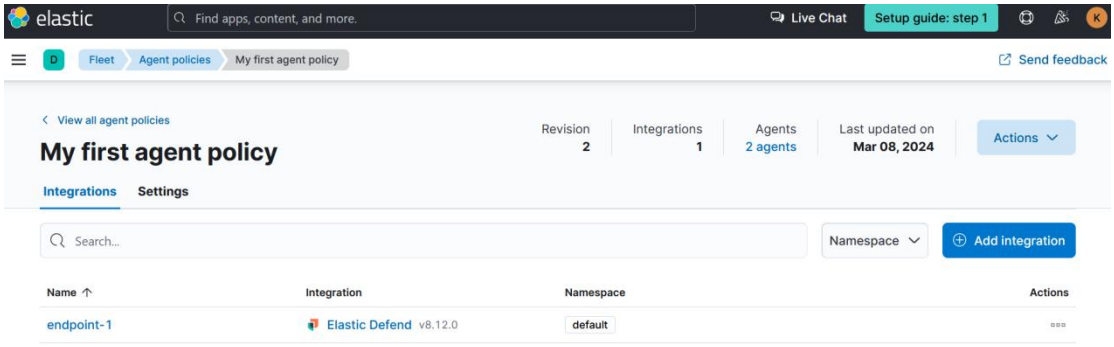


Aquí en el **fleet** vemos las politicas que tenemos:



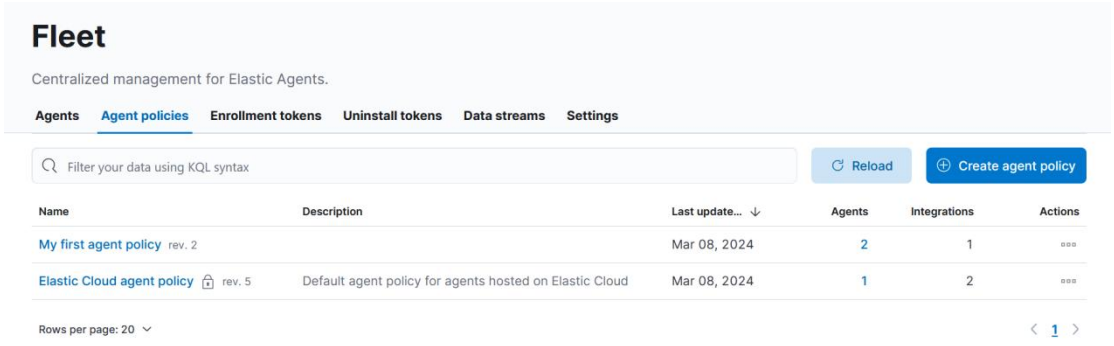
Esto son las partes de las políticas (conjunto de normas que le vamos a poner en nuestros sistemas)

Tanto la kali como el windows lo tenemos en **my first agent policy** (esta política solamente tienen la integración con el Elastic Defend.

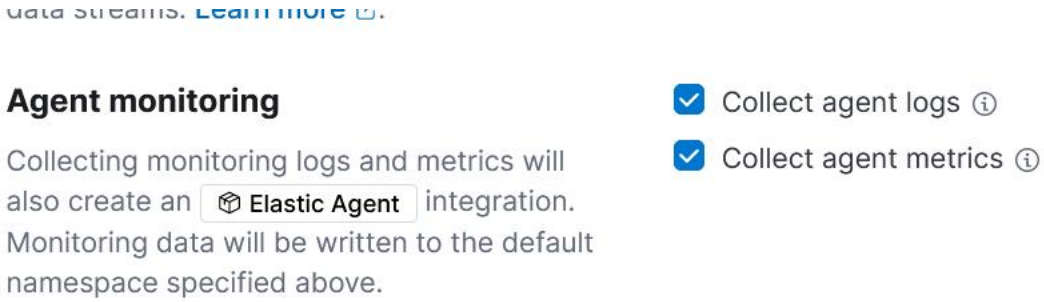


Queremos dividirlo en diferentes politicas para que los windows y los linux nos lo recoja con diferentes pluggins.

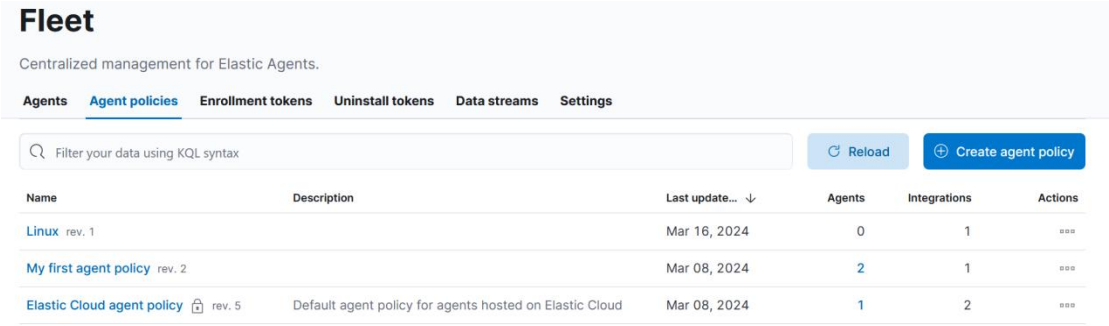
Para ello vamos a **agent policies** y creamos una nueva politica



Aquí vamos a ponerle de nombre linux, y podemos dar a **collect system logs and metrics** para que nos recoja los logs de sistema y métricas de funcionamiento y creamos la política.



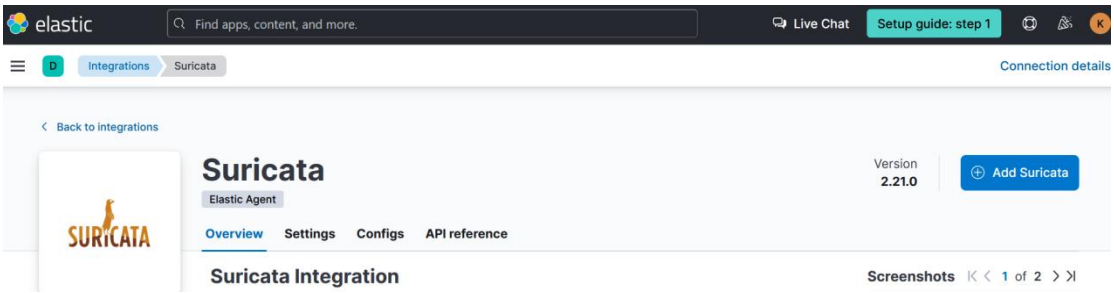
Aquí tendríamos la política linux que hemos creado y la política **my first agent policy.**



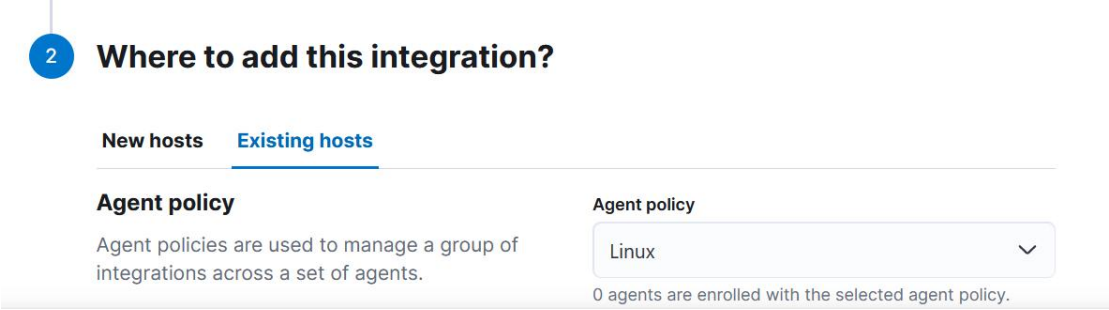
Vamos a añadirle una integracion a la politica de **my first agent policy**, le damos a **add integration**.

Nosotros hemos añadido antes la de Elastic Defend, que es para proteger nuestros host y la que recoge los eventos de seguridad.

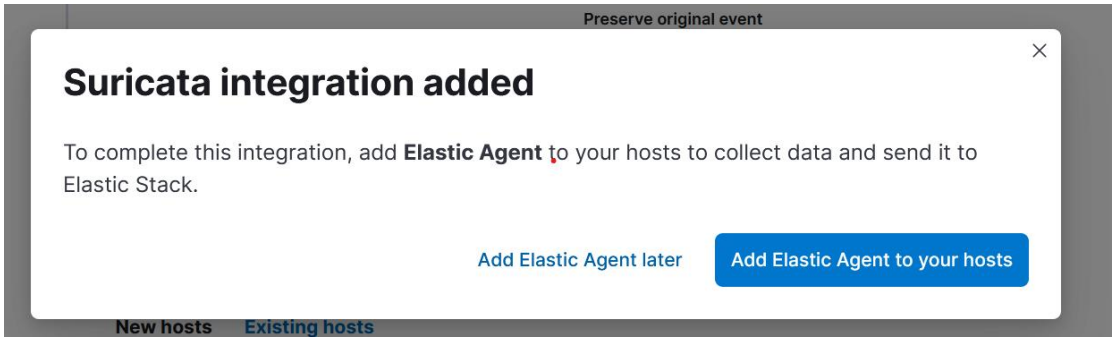
Vamos a añadirle la de **Suricata** para que recoja esos logs. le damos a **add suricata**.



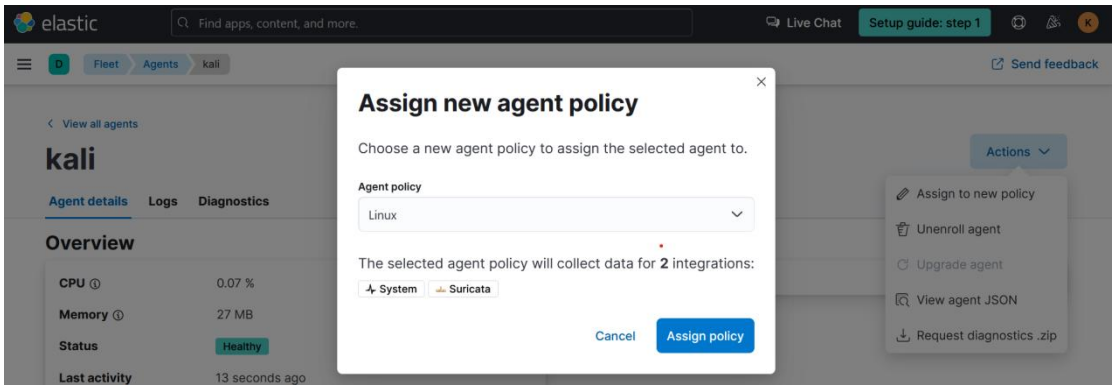
Vamos a ponerla en la política de linux.



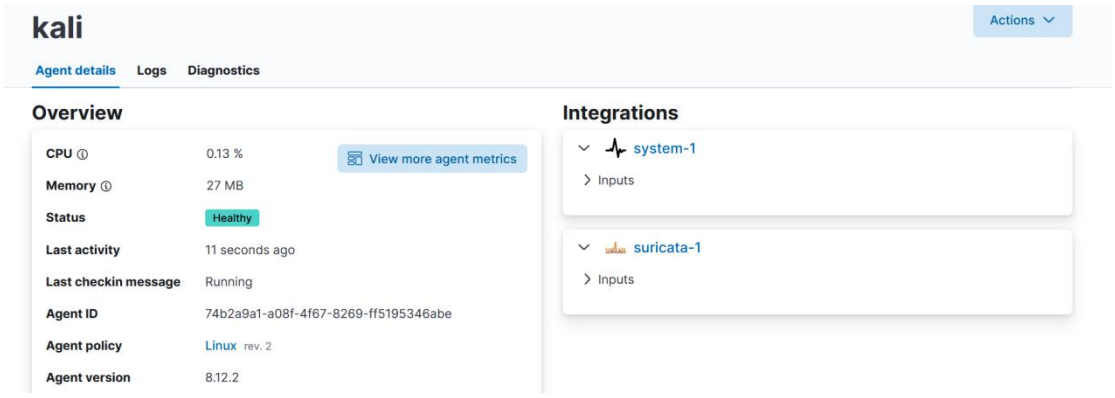
Aquí podemos añadir un agente, pero en principio no vamos a añadir ninguno.



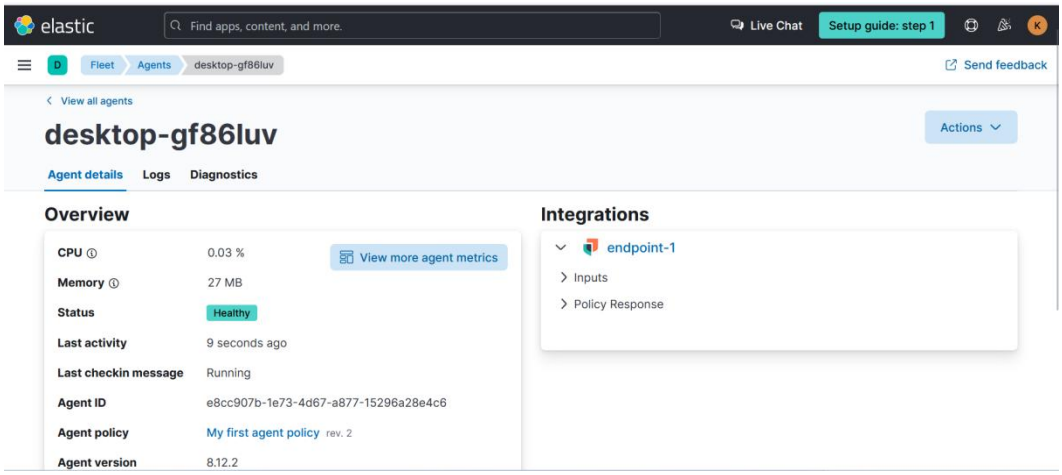
Aquí en el kali, vamos a cambiarle a la política de linux:



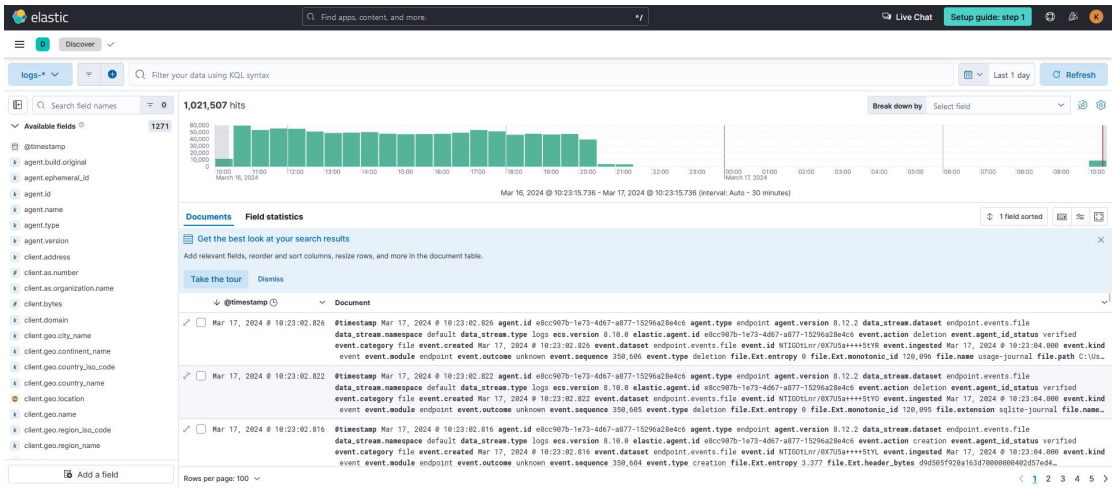
Aquí nos recoge los logs de sistema y los de suricata solamente para el equipo de la kali.



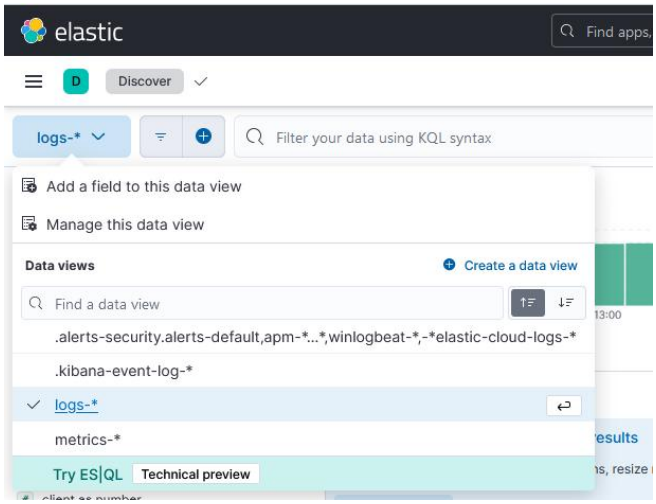
El windows nos está recogiendo logs del end-point.



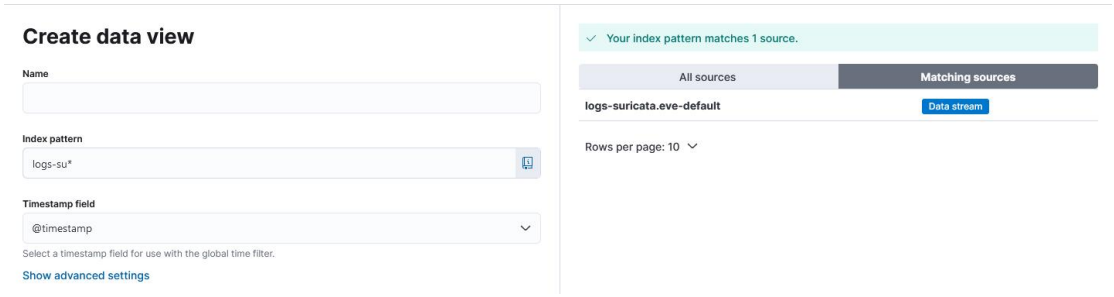
Una vez ya tenemos creadas las políticas, en la parte de D**iscover** nos aparecen los logs que estamos recogiendo.

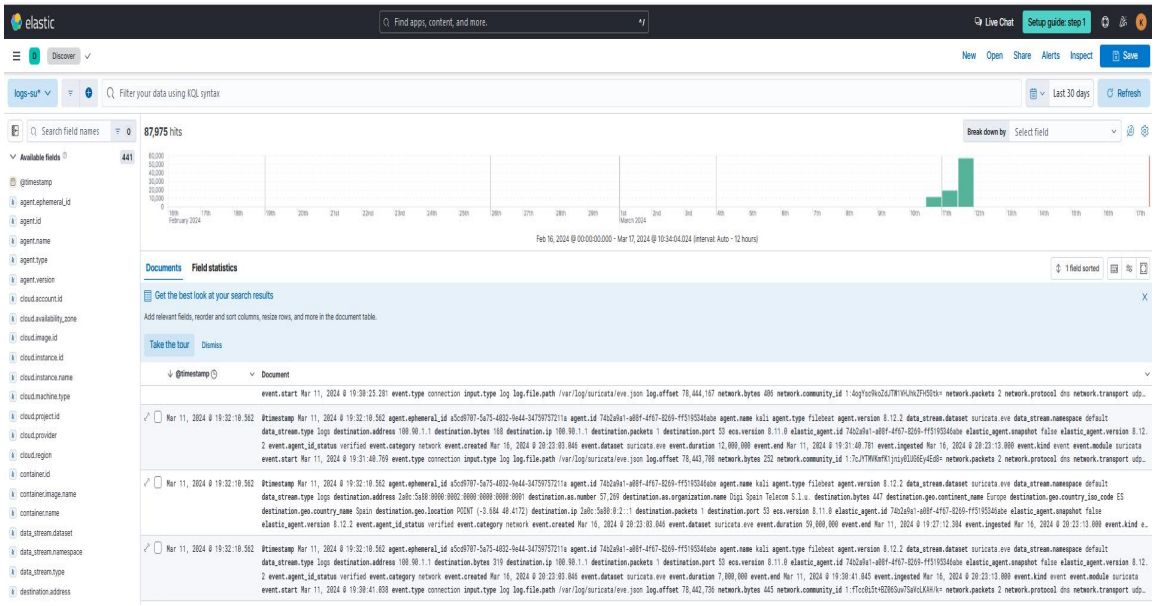


Aquí podemos crear los conjuntos de datos que queremos ver:



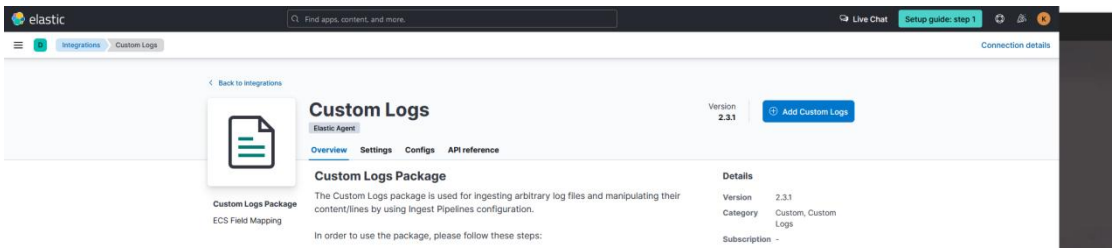
Para ver los logs de Suricata, buscamos **losgs-suricata***, ponemos los ultimos 30 dias y ya podemos ver los eventos de Suricata que están recopilados en el sistema.

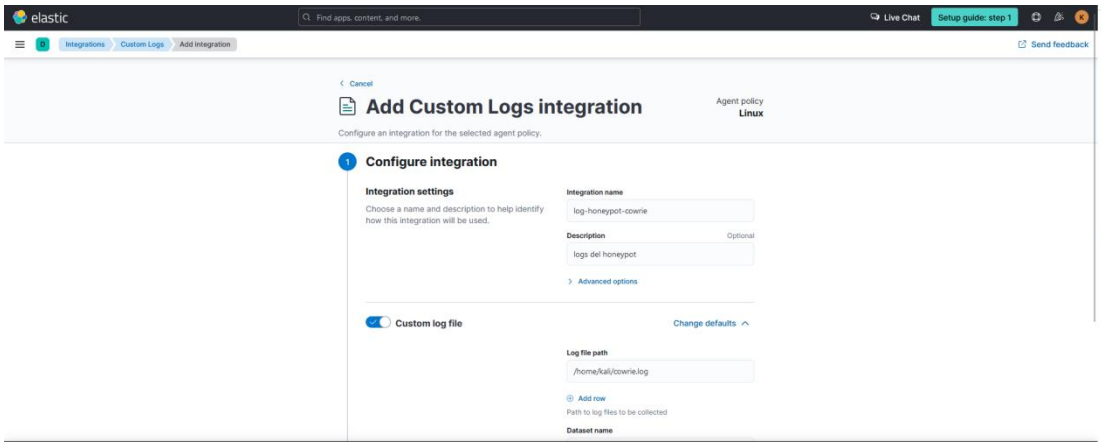


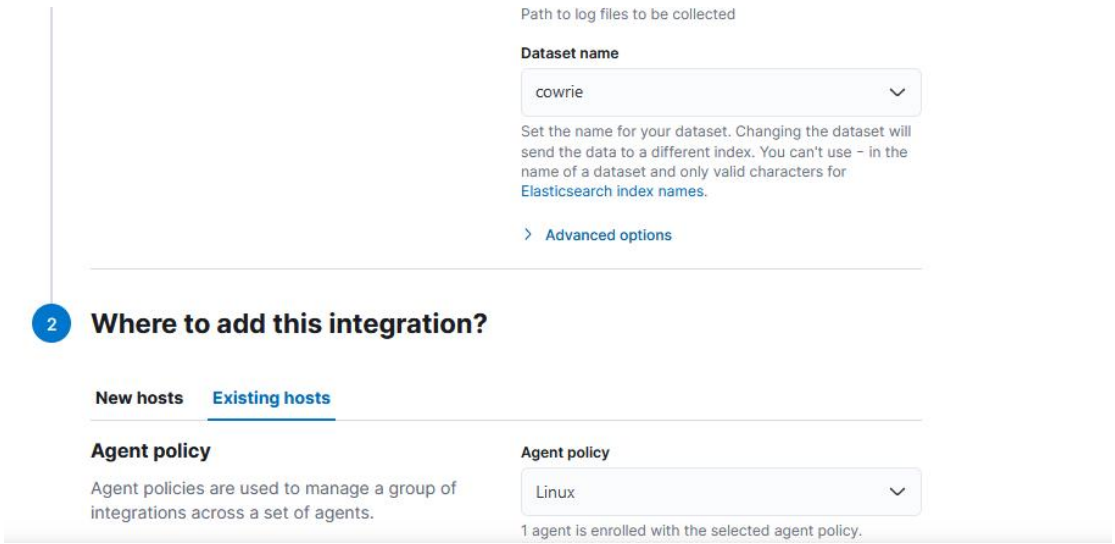


INTEGRACIÓN DE LOS LOGS DEL HONEYPOT EN ELASTIC

Vamos a incluir los logs que hemos recopilado antes en el archivo cowrie.log a Elastic con la ayuda de la integración **Custom Logs**:







Aquí ya podemos ver como nos llegan los logs de nuestro Honeypot al Elastic: