Fuente: Microsoft Copilot

LABORATORIO 7. SEGURIDAD DE RED

DEPARTAMENTO DE INFORMÁTCA. UNIVERSIDAD DE OVIEDO

Seguridad de Sistemas Informáticos | 2024 – 2025 (V4.2 *"S-81 Isaac Peral"*)

Contenido

Infraestructura de este laboratorio

*(**NOTA:** Para evitar problemas, te recomendamos encarecidamente que ejecutes esta infraestructura de laboratorio **en la máquina virtual base en lugar de en sus versiones con hardening** que puedas haber creado en laboratorios anteriores. En la vida real, deberías hacer lo contrario, pero las implementaciones reales también tienen un tiempo de validación para garantizar que todo funcione como se espera después del hardening).*

Este laboratorio **usa la red *bridge* de la máquina virtual** para poder conectar el host a la máquina virtual gracias a la IP asignada en la red a la que ambas se conectan. Sin embargo, hay un problema: algunos servidores DHCP asignan direcciones IP mediante el parámetro VM `machine-id`. Como todas nuestras máquinas se clonan a partir de una base que creamos, **todas tienen el mismo** `machine-id` y, lo tanto, todas podrían recibir la misma IP del servidor DHCP.

Para asegurarse de que esto no pase, de modo que puedas conectarte a la VM de tus compañeros para realizar pruebas y/o evitar errores de red extraños, **genera un nuevo** `machine-id` siguiendo este procedimiento y reinicia la máquina: https://gist.github.com/dllud/0820d53477bce57b7f29d7b8f7761268. Puedes olvidarte del resto de operaciones en este enlace, realmente no las necesitamos. No obstante, también puedes cambiar el nombre de host a uno que te resulte más familiar si quieres.

Por otro lado, **también tendrás que utilizar la "*máquina Wazuh*"** del laboratorio anterior para hacer un ejercicio. Recuerda que si no tienes una máquina en casa capaz de soportar la creación de una VM con estas características, **te recomendamos que priorices *la actividad de Wazuh* para hacerla en el laboratorio**, donde sí tienes *hardware* que soporte esa VM. Incluso puedes empezar con él sin ningún problema.

🔍 ***Este laboratorio también tiene un archivo de .zip adjunto que te permite ejecutar automáticamente PiHole para hacer los ejercicios correspondientes. Descárgalo desde el campus virtual de la VM y descomprímelo***

Bloque 1: Protección de redes de un host. *Firewalls* PF

Ejercicio L7B1_FWIN. Filtrado de tráfico entrante con *UFW*

Descripción de la actividad / Aplicación práctica

Esta actividad está pensada **para enseñaros a usar** `ufw` para realizar operaciones de seguridad típicas relacionadas con la red, pero que implican **el manejo del tráfico entrante**.

Resultados esperados

Esta actividad finalizará cuando **te familiarices completamente con el funcionamiento del firewall** `ufw`, la **prioridad de sus reglas** y cómo se interpretan, lo que le permite bloquear o permitir cualquier servicio de cualquier fuente que desee. Puedes responder a esta pregunta: *¿Sabrías ahora cómo crear una lista de direcciones IP permitidas o una lista de direcciones IP denegadas al establecer conexiones con una máquina?*

❓ ***Pregunta que debe resolver:*** *¿Sabes cómo administrar cualquier operación ufw para denegar/permitir el tráfico entrante, entendiendo al detalle el comportamiento de sus reglas? (especialmente como trata la prioridad de estas)*

Otra información necesaria para su realización

Para realizar esta actividad debes tener `ufw` ya instalado y habilitado (**Lab 1**). La actividad en sí consiste en **usar el *siguiente cheatsheet*** para hacer estas operaciones (en este orden) con el *firewall*. Puede usar la IP de red bridge de la máquina virtual y el equipo físico para probar las operaciones, o hacer equipo con un compañero.

- **Comprueba su estado** para ver los puertos permitidos y denegados en estos momentos.
- Asegúrate de que la **política predeterminada** está configurada para permitir el tráfico saliente y denegar todo el tráfico entrante.
- **Lista las reglas habilitadas** en un formato numerado.
- **Crea reglas que abran los puertos** de algunos servicios conocidos: `telnet`, `ftp`, `www`, `ssh`, `https` para que se pueda acceder a ellos desde cualquier IP.
- **Escanea** la IP de la MV desde el host con `nmap` y mira qué pasa. *¿Se ven los puertos incluso si no hay servicios reales en ejecución detrás?* (abrimos los puertos, pero nunca instalamos servidores de `telnet`, `ftp`, web...)

🔍 ***NOTA: Le pedimos a la administración de la escuela que instalara*** `nmap` ***en las máquinas Windows del laboratorio***

- **Mira el listado de reglas en formato numerado** y usa esta información para eliminar sólo las reglas que pertenecen al servicio `telnet`.
- **Deniega la conexión a cualquier máquina** que pertenezca a tu red de pruebas. Para ello, puedes usar el CIDR de la red bridge y bloquear temporalmente el host, o trabajar con un compañero en el laboratorio.

- **Instala un servidor FTP** en la máquina virtual (`vsftpd`). Intenta conectarse a él desde la máquina que acabas de denegar. *¿El firewall se comporta como esperabas? ¿Por qué?*
- **Habilita las conexiones FTP para que se puedan hacer solo desde el host solo eliminando e insertando reglas en el orden correcto.** *¿Entiendes cómo funciona el orden de las reglas en ufw y, por lo tanto, cómo las procesa este firewall?*
- **Quita las reglas necesarias** para que no queden reglas `DENY`, dejando el *firewall* en un estado adecuado para permitir que el host se conecte de nuevo a la máquina virtual a través de SSH.

ufw 0.36 Cheatsheet (ingenieriainformatica.uniovi.es)

Tool to ease Ubuntu firewall management

https://launchpad.net/ufw

```
root@ssl18base:/etc/wireguard# ufw status
Status: active

To                         Action      From
--                         ------      ----
80/tcp                     ALLOW       Anywhere
443/tcp                    ALLOW       Anywhere
23/tcp                     ALLOW       Anywhere
```

GENERAL USAGE

ufw COMMAND

NOTES

ufw is not enabled by default in a typical Ubuntu installation
To use it you need to enable it first with: sudo ufw enable
It can be disabled at any time with: sudo ufw disable

OPTIONS

allow ARGS: add allow rule
default ARG: set default policy
delete RULE|NUM: delete RULE
deny ARGS: add deny rule
disable: disables the firewall
enable: enables the firewall
insert NUM RULE: insert RULE at NUM
limit ARGS: add limit rule
logging LEVEL: set logging to LEVEL
reject ARGS: add reject rule
reload: reload firewall
reset: reset firewall
route delete RULE|NUM: delete route RULE
route insert NUM RULE: insert route RULE at NUM
route RULE: add route RULE
show ARG: show firewall report
status numbered: show firewall status as numbered list of RULES
status verbose: show verbose firewall status
status: show firewall status
version: display version information

Application profile commands

app default ARG: set default application policy
app info PROFILE: show information on PROFILE
app list: list application profiles
app update PROFILE: update PROFILE

EXAMPLES

NOTE: All examples requires soor privileges (sudo)

DEFAULT BEHAVIOR POLICIES:
* Deny all incoming traffic by default: ufw default deny incoming
* Allow all outgoing traffic by default: ufw default allow outgoing

ALLOW SERVICES:
* sudo ufw allow 22 (or ufw allow ssh)
* ufw allow 'Apache Full' *(there are application profiles available:* sudo ufw app list)
* ufw allow 45/tcp *(allow port and protocol)*
* ufw allow from 192.168.1.1 port 62 *(Source and Destination (allow only from this IP))*
* ufw allow to 127.0.0.2 port 62 *(allow from anywhere to a local interface only)*
* ufw allow 80/tcp comment 'accept Apache' *(comment a rule)*
* ufw allow 1194/udp comment 'OpenVPN server' *(Open UDP/1194 (OpenVPN) server and add a comment)*
* ufw allow 3000:4000/tcp, sudo ufw allow 3000:4000/udp *(allow port ranges; tcp and udp 3000 to 4000)*
* sudo ufw allow from 156.35.94.10 *(allow ALL connections from 156.35.94.10)*
* sudo ufw allow from 156.35.94.10 to any port 22 proto tcp *(allow connections from 156.35.94.10 only to port 22)*
* sudo ufw allow from 156.35.94.10 to 156.35.94.50 port 22 proto tcp *(set destination IP too)*
* sudo ufw allow in on wg0 to any port 22 *(open port 22 for wg0 interface only)*
* ufw allow in on lxdbr0 from 10.100.12.29 to any port 3389 proto tcp *(allow connection for TCP port 3389 on Lxdbr0 interface from 10.100.12.29)*
* ufw allow in on lxdbr0 from 10.100.12.0/24 to any port 3389 proto tcp *(same as previous but allow whole network)*

DENY SERVICES:
* ufw deny 21
* sudo ufw deny 25/tcp *(deny port and protocol)*
* sudo ufw deny from 156.35.94.10 *(deny from specific IP)*
* sudo ufw deny from 156.35.0.0/16 *(deny from specific network, all hosts from the network)*

ENABLE SPECIFIC PROTOCOLS:
* ufw allow to 127.0.0.3 proto esp
* ufw allow to 127.0.0.3 proto ah
To enable IPv6 support, edit /etc/default/ufw and ensure **IPV6=yes**

ENABLE CONNECTION LIMITS:
Allow connections but deny them if an IP attempts 6 or more connections within thirty seconds. I. e.: **sudo ufw limit ssh**

ROUTES (IP Masquerading with ufw):
Edit the /etc/ufw/sysctl.conf and make sure you have the following line not commented: **net/ipv4/ip_forward=1**
* ufw route allow in on eth0 out on eth1 to any port 80 from any *(forward all network requests running on eth1, port 80 to eth0)*

Apply both for incoming and outgoing traffic (bidirectional):
* ufw route allow in on eth0 out on eth1 to 10.0.0.0/8 port 80 from 192.168.0.0/16
* ufw route allow in on eth1 out on eth0 from 10.0.0.0/8 to 192.168.0.0/16

EGRESS FILTERING:
Block RFC1918 addresses (private IPs) going out of eth0 interfaces on your VM connected to the Internet.
* ufw route reject out on eth0 to 10.0.0.0/8 comment 'RFC1918 reject'
* ufw route reject out on eth0 to 172.16.0.0/12 comment 'RFC1918 reject'
* ufw route reject out on eth0 to 192.168.0.0/16 comment 'RFC1918 reject'

LOGGING:
* sudo ufw logging on (enable log)
* sudo ufw logging medium (log verbosity)
By default all UFW entries are logged into the /var/log/ufw.log file

RULE LIST:
* ufw show listening
* ufw show added

🔍 ***Si la línea de comandos no es lo tuyo, puedes hacer este ejercicio con la GUI de UFW (GUFW) siguiendo estas instrucciones: https://www.zdnet.com/article/how-to-add-a-gui-for-your-ubuntu-firewall-and-why-you-should/***

Ejercicio L7B1_FWOUT: Filtrado de tráfico saliente con UFW

Descripción de la actividad / Aplicación práctica

Esta actividad está pensada para enseñarnos a usar `ufw` para realizar operaciones de seguridad típicas relacionadas con la red, pero que implican **la administración del tráfico saliente**.

Resultados esperados

Esta actividad finalizará cuando puedas **restringir el tráfico saliente a voluntad** , excepto para los servicios que quieras, y comprendas los beneficios de seguridad (y los problemas de uso) que esto puede traer. *¿Entiendes ahora la diferencia entre el tráfico saliente y el entrante?*

***Pregunta que debes resolver**: ¿Sabe cómo administrar cualquier operación ufw para denegar/permitir el tráfico saliente, entendiendo completamente el comportamiento de sus reglas?*

Otra información necesaria para su realización

Esta actividad consiste en ampliar la anterior para **prohibir todo el tráfico de salida** excepto el correspondiente al servicio de actualización de paquetes (`apt`) y entender qué significa esto. Ten en cuenta que `apt` se conecta a Internet utilizando los protocolos `http` y `https`. Elimina esta regla una vez la pruebes.

Una vez termines, habilite otra regla para bloquear las conexiones desde cualquier puerto a la red `156.35.0.0/16` (red de Uniovi) y prueba que funciona. Elimina esta regla una vez termines de probarla. Puedes usar el *cheatsheet* anterior (y/o GUFW) para ayudarte con este trabajo.

Bloque 2: Protección de red en conexiones salientes y entrantes

Ejercicio L7B2_DNSPI: Protección de conexiones salientes: Filtrado de DNS con *PiHole*

Descripción de la actividad / Aplicación práctica

Necesitas **bloquear conexiones a dominios maliciosos** conocidos usando un DNS local que filtre las peticiones que tu máquina hace a dominios externos

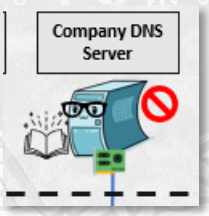
Resultados esperados

Esta actividad finalizará cuando puedas **configurar el servidor de filtrado DNS *PiHole*** para probar cómo cambia tu navegación una vez que uses sus funciones. Puedes contestar a las preguntas que se hacen a lo largo del ejercicio y esta otra: *¿Cuál crees que es la diferencia más significativa entre PiHole y el NextDNS que veras luego?*

❓ ***Pregunta que debes resolver**: ¿Sabes cómo gestionar tu propio DNS con funciones de seguridad gracias al software PiHole?*

Otra información necesaria para su realización

Te recomendamos encarecidamente que crees una instantánea de la máquina virtual antes de realizar este ejercicio. El objetivo de este ejercicio es mostrarte la importancia de controlar tu propio servidor DNS para filtrar el tráfico saliente potencialmente malicioso. Esto corresponde a esta parte de la teoría *Secure Network Infrastructure* (SNI), **pero también lo puedes usar en casa** 😊.



🔍 ***Haremos esto configurando un producto profesional, PiHole (https://pi-hole.net/), que te permite tener un servidor DNS que además filtra cualquier solicitud que reciba utilizando un enfoque blocklist: cualquier conexión realizada a una IP presente en la lista de bloqueo se anulará.***

PiHole permite la instalación en muchos sistemas, pero en este laboratorio lo tenemos empaquetado en un contenedor. Vete a la carpeta `pihole` de los archivos de laboratorio y ejecuta el script `run_pihole.sh`. Una vez termine de ejecutarse el comando, vete a `127.0.0.1/admin` en el navegador de la VM para ver la pantalla de estadísticas de bienvenida de *PiHole*. La contraseña de administrador es `test123...`

🔍 ***Por favor, asegúrate de que Apache 2 (o cualquier otro servidor web) no esté instalado en la máquina virtual y haz*** `sudo apt-get remove apache2` ***si lo está. No debería, pero nunca se sabe si ha quedado de alguna prueba anterior***

Te recomendamos que navegues ahora a cualquier página web con publicidad (como www.elmundo.es) y dejes esa web abierta. Ahora que *PiHole* está funcionando, para empezar a filtrar conexiones solo tenemos que cambiar el DNS de nuestra tarjeta de red `enp0s3` (la que se conecta a Internet) a `127.0.0.1` (*PiHole* se ejecuta localmente).

🔍 ***No te preocupes, no perderás la conexión a Internet, PiHole sabe qué hacer*** 😊

La siguiente imagen muestra los cambios que debes realizar para esto (por favor, hazlo sobre la tarjeta de red que es la que usas para salir a Internet). **También tendrías que cambiarlo en el fichero** `/etc/resolv.conf` si quieres hacer el cambio permanente, pero en este caso no es necesario.

```
 GNU nano 2.9.3            /etc/netplan/01-netcfg.yaml            Modified

1 network:
2   version: 2
3   ethernets:
4     eth0:
5       dhcp4: true
6       nameservers:
7                 addresses: [127.0.0.1]
```

Vuelve a cargar ahora el sitio web anterior. *¿Ves alguna diferencia? ¿Qué pasará ahora con cualquier dispositivo que utilice este servidor DNS si, por ejemplo, tu router te dejase instalarlo (algunos lo hacen, o tienen algo muy similar integrado)?*

🔍 ***NOTA: Las siguientes veces que rearranques la MV, PiHole seguirá funcionando. No obstante, salvo que hagas el cambio del DNS en*** `/etc/resolv.conf` ***seguirá usando el DNS original de la máquina, no PiHole, aunque este esté activo. No afectará a tu trabajo con la asignatura, pero es necesario que lo sepas*** 😊

🛠 Ejercicio L7B2_DNSPIPLUS: Filtrado avanzado de conexiones salientes con *PiHole*

Descripción de la actividad / Aplicación práctica

Necesitas bloquear conexiones a aún más dominios maliciosos, **potenciando las capacidades de *PiHole***

Resultados esperados

Esta actividad finalizará cuando puedas **mejorar las listas de bloqueo de *PiHole*** y bloquear así conexiones a más sitios web maliciosos conocidos.

❓ ***Pregunta que debes resolver****: ¿Sabes cómo potenciar las funciones de bloqueo de PiHole?*

Otra información necesaria para su realización

Aunque *PiHole* filtra un gran número de sitios web maliciosos de serie, podemos darle más "potencia" utilizando las URL maliciosas recopiladas por ***The Block List Project*** (https://blocklistproject.github.io/Lists/). Sigue estas instrucciones para incorporar esta lista de IPs a *PiHole*:

- Copia el enlace de la lista más completa (***Everything***) en formato compatible con *PiHole* (https://blocklistproject.github.io/Lists/everything.txt).
- Agrega la URL a las listas de bloqueo de tu *PiHole* (***Login - Group Management-Adlists -Pega la URL de la lista en el campo "Dirección", agrega un comentario - Haz clic en "Add"***)
- Actualiza ***Gravity*** (***Tools-Update Gravity-Click "Update"***) para incorporar las nuevas URL a la lista de bloqueo. El proceso lleva un tiempo, **¡por favor espera y no navegues fuera de esta página!**

Navega de nuevo a cualquier web con la página web de estadísticas *de PiHole* abierta y observa cómo están ahora en funcionamiento listas de bloqueo mucho más grandes.

Ejercicio L1B3_NEXTDNS: Usar un DNS seguro para tu protección: *NextDNS*

Descripción de la actividad / Aplicación práctica

Puedes usar el servicio *NextDNS* de forma gratuita, incluso sin crear una cuenta, para poder navegar de manera mucho más segura de manera transparente, para entender la enorme importancia del servicio DNS en tu navegación.

Resultados Esperados

El objetivo de este ejercicio es que te des cuenta de que **sin un servidor DNS** que resuelva tus peticiones a IPs realmente **no podrías navegar**. Por ello, si ese servicio de resolución **tiene "inteligencia" que impida navegar a webs que se sepan que son perjudiciales**, podríamos lograr una web más segura (aun con un riesgo de censura). *NextDNS* es esto, pero que además te permite controlar esa "inteligencia". Una vez lo pruebes, la idea es que puedas contestar estas preguntas:

- *¿Entiendes que al usar NextDNS estás poniendo en marcha una gran cantidad de medidas de seguridad de forma transparente (sin que seas consciente de ello)?*
- *¿Qué servicios de los que proporcionan NextDNS crees que te pueden resultar más útiles?*
- *¿Has notado alguna diferencia a la hora de navegar (errores, velocidad...) o algún problema con alguna página web?*
- *A la vista de tu experiencia con el servicio, ¿Lo instalarías en tu casa? ¿Lo harías en una máquina virtual que tengas para "navegación segura" junto con otras medidas de seguridad?*

❓ ***Pregunta que debes resolver**: ¿Puedes configurar NextDNS para tener la máxima protección de navegación posible? ¿Y para bloquear un tipo de páginas web? ¿O un conocido videojuego?*

Otra información necesaria para su realización

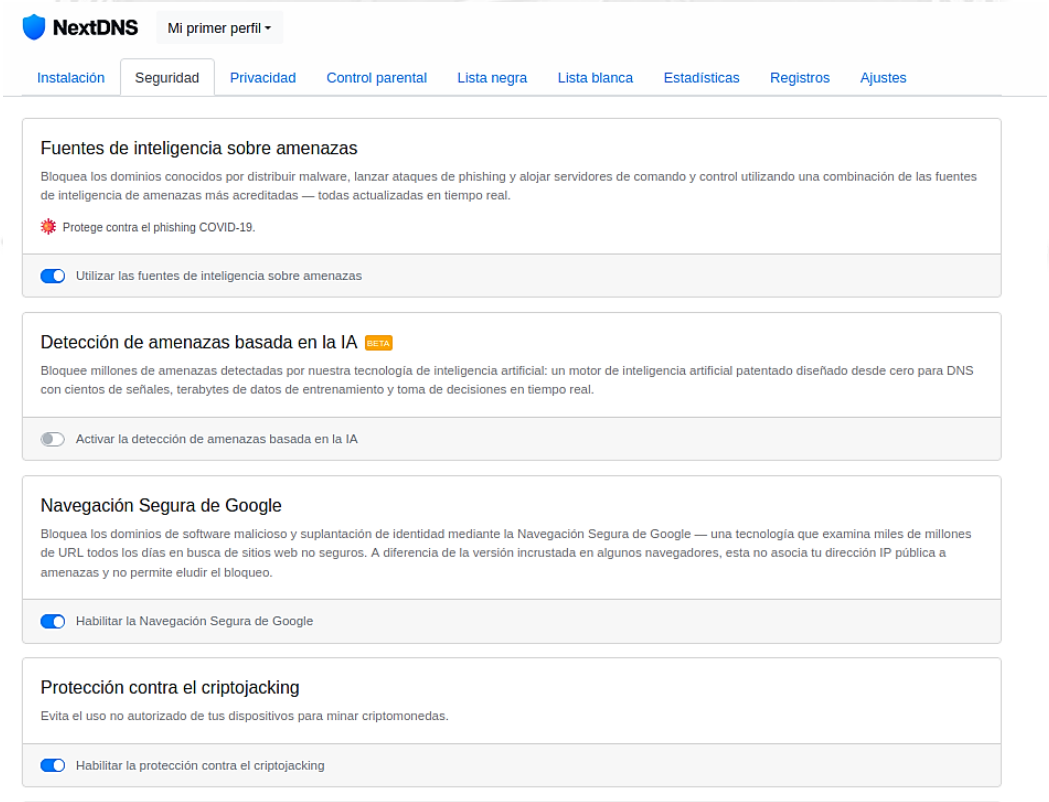
El servicio *NextDNS* (https://nextdns.io/) realmente es una forma de librarnos de muchos problemas de una manera transparente, y que se puede usar para tener una navegación mucho menos propensa a problemas.

Una vez se configure en los navegadores o máquinas, el servicio se encargará automáticamente de **bloquear cualquier tipo de acceso bien directo o indirecto** (porque otra página lo cargue) a dominios que se consideren amenazas, o que pertenezcan a empresas anunciantes, que no estén aprobadas por este servicio centralizado de DNS.

🔍 ***La ventaja de este servicio es que ni siquiera te tienes que hacer una cuenta para usarlo, y si te la haces es para guardar tu configuración particular. Tampoco hace falta instalar nada si no quieres. Si bien el servicio gratuito está limitado a 300000 peticiones al mes, después de las cuales seguirá sirviéndote páginas web pero no tendrás protección, ese volumen de peticiones es suficiente para una navegación normal de cualquier usuario de conocimientos básicos.***

Para usar este servicio tenemos que ir a su web y usar la opción "***Try it now***" de su página principal, tras lo cual iremos a una URL que tiene este aspecto: `https://my.nextdns.io/<código de letras y nºs aleatorio>/setup`. En esa URL podremos configurar las opciones de seguridad del servicio y ver las opciones de instalación en nuestra máquina si finalmente queremos usarlas.

Como se ve, en la primera pestaña de ***Seguridad*** tenemos activas una serie de medidas por defecto, y se recomienda activarlas todas inicialmente a ver cómo se comporta con nuestra navegación habitual:



Hay un gran nº de medidas que podemos activar, por lo que se recomienda recorrer toda la sección y activar todas inicialmente. Ten en cuenta que cada medida tiene una explicación de lo que es exactamente. Aunque no la entendamos, se recomienda su activación, y sólo desactivarlas si hay algún problema en nuestro uso habitual. El ejercicio consiste en **recorrer todas las opciones** que nos da *NextDNS* y activar todas las que creamos útiles para nuestro caso de uso particular:

- La opción de **bloquear TLDs** sirve para bloquear el acceso a páginas cuyo dominio acabe en un prefijo concreto (`.es`, `.com`, etc.). Puedes encontrar una lista completa aquí: https://en.wikipedia.org/wiki/List_of_Internet_top-level_domains.
- En el apartado de **privacidad** podemos encontrar una característica muy útil: **las listas de bloqueo**. Son básicamente listas de dominios que el servicio no resolverá por pertenecer a anuncios, páginas

maliciosas o servicios intrusivos o sin interés que se han considerado perjudiciales de alguna forma. Aunque el servicio viene con una ya cargada, podemos mejorarlo usando listas que están disponibles por Internet creadas para *NextDNS*. En este enlace puedes encontrar algunas: https://github.com/hagezi/dns-blocklists. Esto es muy similar a las características que hemos visto de *PiHole*.

- *NextDNS* tiene también un apartado de **control parental** donde puedes limitar la búsqueda y el uso de Internet a las horas y dominios clasificados que quieras. Si vas a usar esta opción, ten en cuenta que implica que las páginas web a las que navegues **deben estar clasificadas por NextDNS dentro de la categoría a la que pertenecen**, o especificarla (y no todas lo hacen).
- El **límite de acceso a aplicaciones y juegos** consiste en seleccionar algunos de una lista que el servicio tiene y que queramos prohibir.
- **El resto del servicio** son listas de bloqueo de sitios concretos que queramos, listas para permitir sitios sin importar que encajen en alguna restricción que hayamos puesto, y estadísticas de bloqueo y uso del servicio para que lo tengamos controlado.
- En el apartado de **instalación** tenemos instrucciones para poner en marcha el servicio en cualquier sistema operativo (de PC o de móvil), navegadores concretos (si no queremos hacerlo para todo el sistema operativo) o incluso en ***routers***, pero solo modelos concretos que permitan la ejecución de los comandos que nos indican.

Ten en cuenta que **se genera una nueva configuración** (distinto bloque de nºs y letras en la URL) cada vez que entras en el servicio con el botón para probarlo que hemos visto. **Apunta cuál es tu identificador** para poder seguir configurando el servicio más tarde, y ten en cuenta que la configuración en los navegadores o SO usan ese identificador para distinguir tu configuración de la de los demás.

🔍 ***Si quieres empezar por algo sencillo para probar qué tal funciona, lo mejor es configurar uno de tus navegadores solamente con la configuración que has puesto y ver qué tal va***

Ejercicio L7B2_FAIL2BAN: Protección de intentos de intrusión: *Fail2Ban*

Descripción de la actividad / Aplicación práctica

Necesitas **bloquear las conexiones de las máquinas** que tratan de acceder a tu servicio SSH sin éxito múltiples veces

Resultados esperados

Esta actividad finalizará cuando puedas proteger [redacted]tilizando el modo de protección agresivo de `fail2ban`. También puedes verificar que la protección funciona y revocar los bloqueos de IP creados por este programa. Puedes contestar a esta pregunta: *¿Qué tipo de ataque sobre SSH crees que bloquea esta herramienta especialmente?*

❓ ***Pregunta que debes resolver:*** *¿Sabes cómo habilitar y probar la protección de Fail2Ban?*

Otra información necesaria para su realización

Fail2Ban es una aplicación que **procesa los *logs* del sistema** para buscar indicios de ataques. Cuando se detectan, de acuerdo con las reglas que se le definan, `fail2ban` agrega una nueva regla al *firewall* que

bloquea la IP desde la que se hizo el intento de ataque identificado. Esta regla bloquea la IP del atacante de forma temporal o permanente, dependiendo de la configuración.

🔍 ***También se puede notificar por correo electrónico a un usuario que un ataque está en marcha (instalando `sendmail`, pero no lo usaremos en este laboratorio).***

`fail2ban` se usa típicamente para **proteger** `ssh` pero tiene varias reglas predefinidas por defecto (llamadas ***jails*) que también funcionan con otros servicios, como *Apache 2 o Nginx* (servidores web).**

🔍 ***Técnicamente, cualquier servicio de red que genere un log puede ser protegido por `fail2ban` siempre que tenga configuradas las reglas adecuadas. Para hacer esto puedes crear tus propias reglas, pero eso excede el propósito de este laboratorio***

Para completar este ejercicio, debes **buscar cómo hacer e implementar** estas tareas:

- Bloquear las conexiones `ssh` entrantes **desde el host a la máquina virtual** si no puedes conectarte con una contraseña correcta **más de 3 veces en 60 segundos**. El bloqueo debe durar 30 minutos.
- **Comprobar** que el bloqueo se aplica **provocando la condición** que desencadena `fail2ban`
- **Comprobar el estado de `fail2ban`** para ver que la conexión está registrada correctamente como bloqueada
- **Quitar el bloqueo**

Ejercicio L7B2_PORTSENTRY: Protección de intentos de escaneo: *PortSentry* como *honeypot*

Descripción de la actividad / Aplicación práctica

Necesitas bloquear conexiones de atacantes que **están intentando escanearte**

Resultados esperados

Esta actividad finalizará cuando seas capaz de **proteger un host contra los intentos de escaneo** con `nmap` usando `portsentry`, y comprender el comportamiento del bloqueo que realiza, todo sin interferir con el *firewall* de la máquina. También debes **verificar que la protección funciona** y **deshacer los bloqueos** de IP creados por este programa.

❓ ***Pregunta que debes resolver:*** *¿Sabes cómo habilitar un honeypot simple para nmap que bloquee las máquinas que intentan escaneos?*

Otra información necesaria para su realización

Los escaneos de puertos con *Nmap* se tratarán en profundidad el próximo laboratorio, y son una de las operaciones más comunes en cualquier actividad de *pentesting*. En este laboratorio veremos cómo detenerlos usando esta herramienta. ***PortSentry* puede detectar escaneos de puertos y bloquearlos**, por lo que la máquina que está escaneando no obtendrá información.

Existen varias formas de utilizar `portsentry`, pero vamos a usar **una de las más interesantes**: combinarlo con el *firewall* que desplegamos en una actividad anterior para **emular el comportamiento de un *Honeypot***. Para ello, **dejaremos algunos puertos abiertos a propósito en el *firewall***, aunque no haya ningún servicio

que los escuche. Se detectarán los intentos de escaneo a través de estos puertos "falsamente abiertos " y la herramienta procederá a prohibir la máquina que hace el escaneo. Ten en cuenta que `portsentry` no puede escuchar en los puertos que tienen servicios reales que los están usando, ya que "ocupan" los puertos. Sigue estos pasos:

- Instala el servicio *PortSentry* en tu máquina virtual *Ubuntu* y comprueba que se está ejecutando
- Entiende los **dos modos de trabajo** de la herramienta (valores de los parámetros `TCP_MODE` y `UDP_MODE` en el archivo `/etc/default/portsentry`)
    - **Modo básico** (valores: `"tcp"`, `"udp"`): Escucha una lista estática de puertos predefinidos en el archivo de configuración. **No** utilizaremos este modo ya que la presencia de la herramienta se puede detectar fácilmente.
    - **Modo *stealth* avanzado** (valores: `"atcp"`, `"audp"`): Usa un *raw socket* para detectar escaneos y así la herramienta no puede ser detectada fácilmente: los puertos parecen tener el servicio "normal" en él, pero es `portsentry` quien está escuchando y detectando actividad sospechosa.
- Entiende las **posibles respuestas** a un intento de escaneo (valores de `BLOCK_UDP` y `BLOCK_TCP` en el archivo `/etc/portsentry/portsentry.conf`)
    - Solo hacer **log** del tráfico sospechoso en `/var/log/syslog` (valor: `"0"`) (**valor por defecto**)
    - **Bloquear a la máquina "infractora"** (valor: `"1"`)
    - **Ejecutar un programa** como respuesta (valor: `"2"`). Esto nos da mucha flexibilidad y permite respuestas "salvajes" a los intentos de escaneo 😉, pero no lo usaremos aquí.

Una vez que entiendas esto, haz las siguientes tareas:

- Habilitar algunos servicios en el *firewall*, incluso si no están instalados, como hicimos en el primer ejercicio
- Una vez hecho esto, abre el fichero `/etc/default/portsentry` y pon `TCP_MODE` a `"atcp"` y `UDP_MODE` a `"audp"` **para habilitar el modo *stealth avanzado***.
- Reinicia `portsentry` y haz un análisis `nmap` simple desde tu host a la MV
- Una vez finalizado el análisis, comprueba el contenido de `/var/log/syslog` para ver si se han detectado los intentos de análisis.

🔍 ***Que se haga log con éxito de los intentos de escaneo es la condición para terminar de configurar `portsentry` según lo previsto***

- Abre `/etc/portsentry/portsentry.conf` y cambia los valores de los parámetros `BLOCK_UDP` y `BLOCK_TCP` a `"1"`. Reinicia `portsentry` y vuelve a realizar el mismo análisis. *¿Se ha bloqueado tu host para todos los servicios? ¿Puedes hacer `nmap` a la MV otra vez?*

Revertir bloqueos

Para revertir un bloqueo, es necesario hacer esto:

- Eliminar la IP de la máquina bloqueada de `/etc/hosts.deny`. **Estar en este archivo impide cualquier comunicación desde esta IP a nuestra máquina**.

```
  GNU nano 2.9.3                    /etc/hosts.deny

# /etc/hosts.deny: list of hosts that are _not_ allowed to access the system.
#                  See the manual pages hosts_access(5) and hosts_options(5).
#
# Example:    ALL: some.host.name, .some.domain
#             ALL EXCEPT in.fingerd: other.host.name, .other.domain
#
# If you're going to protect the portmapper use the name "rpcbind" for the
# daemon name. See rpcbind(8) and rpc.mountd(8) for further information.
#
# The PARANOID wildcard matches any host whose name does not match its
# address.
#
# You may wish to enable this to ensure any programs that don't
# validate looked up hostnames still leave understandable logs. In past
# versions of Debian this has been the default.
# ALL: PARANOID

ALL: 192.168.8.100 : DENY
```

- Eliminar la regla del *firewall* que rechaza (`REJECT`) todo el tráfico del cliente infractor: `sudo route del -host <IP del cliente infractor> reject`
- Podemos usar `netstat -rn` para comprobar que se ha eliminado la regla.

Bloque 3. Detección de intrusiones

Ejercicio L7B3_IDSMALTRAIL: Instalación y prueba del IDS *Maltrail*

Descripción de la actividad / Aplicación práctica

Esta actividad te enseña a **instalar y usar un IDS llamado *Maltrail***, para que veas cómo funciona un IDS de red o NIDS.

Resultados esperados

Esta actividad se considerará completa cuando tengas **una instalación operativa de *Maltrail*** capaz de detectar incidencias como escaneos de puertos. Puedes contestar a esta pregunta: *¿Entiendes como un IDS bien configurado se puede usar para localizar actividades sospechosas que vengan de máquinas ajenas?*

❓ ***Pregunta que debes resolver**: ¿Puedes instalar, ejecutar y probar el IDS MalTrail?*

Otra información necesaria para su realización

En esta actividad vamos a ver **un IDS sencillo que tiene una interfaz de uso simple** y cumple con la misma función: *Maltrail*. *Maltrail* es una herramienta de detección de tráfico malicioso autónoma que **se ejecuta en un nodo de monitorización** por el que se supone que pasa todo el tráfico de una red (recuerda qué tipo de máquinas cumplen con esto en nuestra SNI de teoría, como puede ser un ***proxy* inverso**).

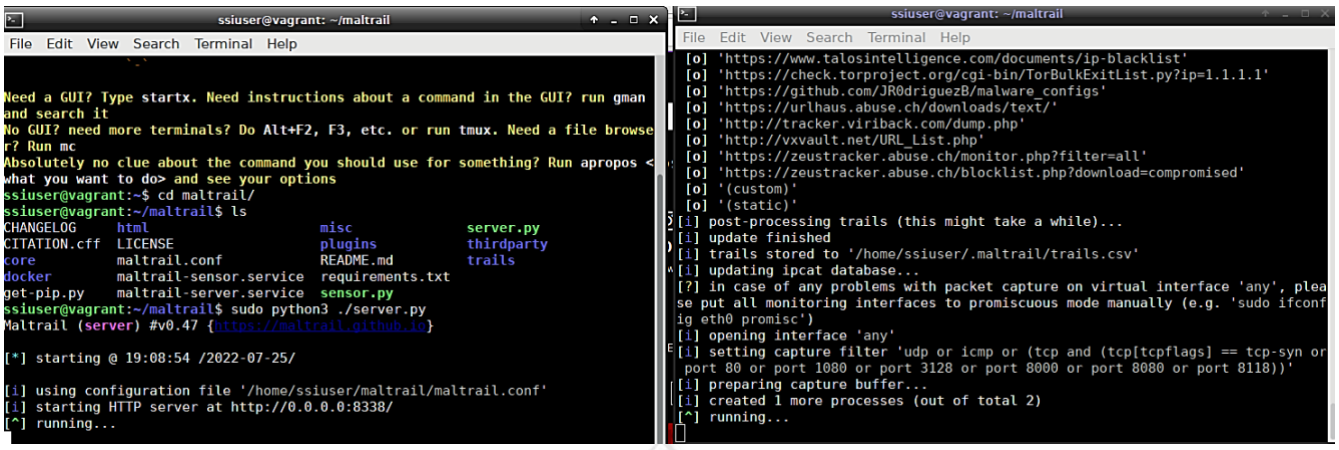
Este IDS usa listas de bloqueo públicas de direcciones web sospechosas y/o *malware*, junto con las direcciones estáticas recopiladas por diversos informes antivirus para detectar posibles amenazas. En caso de que se detecten, los detalles del evento se envían al servidor y se almacenan en un *log*. No obstante, esto ocurre si el componente llamado "**sensor**" se ejecuta en una máquina distinta que el componente llamado "**servidor**".

🔍 ***Para simplificar ambos componentes pueden residir en la misma máquina, como haremos en esta actividad***

Maltrail se puede instalar en cualquier máquina *Linux*, y para ello lo mejor es seguir su tutorial de instalación oficial: https://github.com/stamparm/maltrail. No obstante, ten en cuenta lo siguiente:

- Necesitas tener instalado el paquete `pcapy`. Puedes instalarlo con `sudo apt install python3-pcapy`
- El usuario y la contraseña por defecto se encuentran en la página oficial (`admin`/`changeme!`)

Terminada la instalación de ambos componentes en la misma máquina, y tal y como se indica en el tutorial, arrancamos el sensor y el servidor:



Y con esto ya podremos acceder a http://localhost:8338 para acceder al interfaz. Para probar el IDS basta con hacer un escaneo con `nmap` desde el *host* a la máquina virtual que tiene *Maltrail* operativo y, **tras recargar su interfaz web** debería detectar un posible escaneo de puertos como actividad maliciosa.

🔍 ***Detecta más cosas, pero un escaneo de nmap es lo más sencillo en nuestra LAN. Si lo dejáis expuesto a Internet os podéis asustar de la de cosas que pueden aparecer ahí simplemente por estar conectado...***

BACK
1
2
3
4

Ejercicio L7B3_WAZUHSCAN: *Wazuh* y ataques de red

Descripción de la actividad / Aplicación práctica

Consiste en ver **cómo reacciona *Wazuh*** ante un escaneo de red o un intento de acceso vía SSH fallido

Resultados esperados

Puedes ver cómo reacciona el XDR *Wazuh* de actividades anteriores cuando se somete a los casos cubiertos por ***Fail2Ban*** (intento de acceso repetido fallido mediante SSH) y ***PortSentry*** o ***Maltrail*** (escanear puertos con `nmap`). Puedes responder a estas preguntas:

- *¿Bloquea Wazuh estas actividades o solo informa de que las ha detectado?*
- *¿Detecta Wazuh las actividades mencionadas igual que el resto de las aplicaciones específicas o lo hace de manera menos evidente?*
- *¿Qué diferencia ves entre la información que te da Maltrail y Wazuh tras un escaneo con `nmap`?*
- *¿Entiendes por tanto mejor ahora la diferencia entre un NIDS y un HIDS?*

Otra información necesaria para su realización

Esta actividad es sencilla, puesto que solo tienes que **arrancar la "*MV Wazuh*"** de actividades anteriores y, desde tu host hacer un `nmap` e intentar **acceder varias veces vía SSH** con usuario / password erróneas a la MV de la asignatura. Hecho esto, examina la información capturada por *Wazuh* para poder responder a las preguntas realizadas en el ejercicio.

Bloque 4. Extremando la seguridad en las conexiones: VPNs

Ejercicio L7B4_VPNP2P: Uso de VPNs P2P: *ZeroTier*

Descripción de la actividad / Aplicación práctica

Consiste en usar la VPN P2P comercial *ZeroTier* en su versión gratuita para **familiarizarse con un servicio VPN P2P y las diferencias principales con una tradicional.**

Resultados esperados

Puedes **poner en marcha una VPN P2P *ZeroTier* entre dos MVs** y comprobar que ambas están comunicadas entre ellas como si fuera una red local. Puedes hacer esto **con dos MVs tuyas o con la de un compañero**. Puedes contestar a estas preguntas y a las que se plantean a lo largo del ejercicio:

- *¿Entiendes como este sistema te da el control de la VPN y no dependes tanto de un servidor como en el caso de ProtonVPN?*
- *¿Entiendes cómo puedes usar este servicio para conectarte de forma segura desde un PC del laboratorio a tu casa?*
- *¿Entiendes que aunque no tengas permisos de instalación de programas en un PC de un laboratorio, los tienes en la MV, y puedes comunicar por tanto la MV con el PC de tu casa sin problemas si ambos tienen cliente de ZeroTier?*
- *¿Se te ocurre como combinar el uso de un firewall con ZeroTier para que solo puedas acceder a los servicios desde la red que pones en marcha con ZeroTier?*
- *¿Entiendes ahora que puedes dejar servicios (Escritorio Remoto, carpetas compartidas...) en una máquina de tu casa sin exponer en Internet, solo accesibles para las máquinas detrás de tu router (no abres puertos en el router para ellos nunca, por tanto no son visibles "desde fuera"), y que gracias a ZeroTier puedes acceder a ellos desde cualquier parte?*

❓ ***Pregunta que debes resolver**: ¿Se puede crear una conexión VPN P2P entre dos hosts o VM diferentes?*

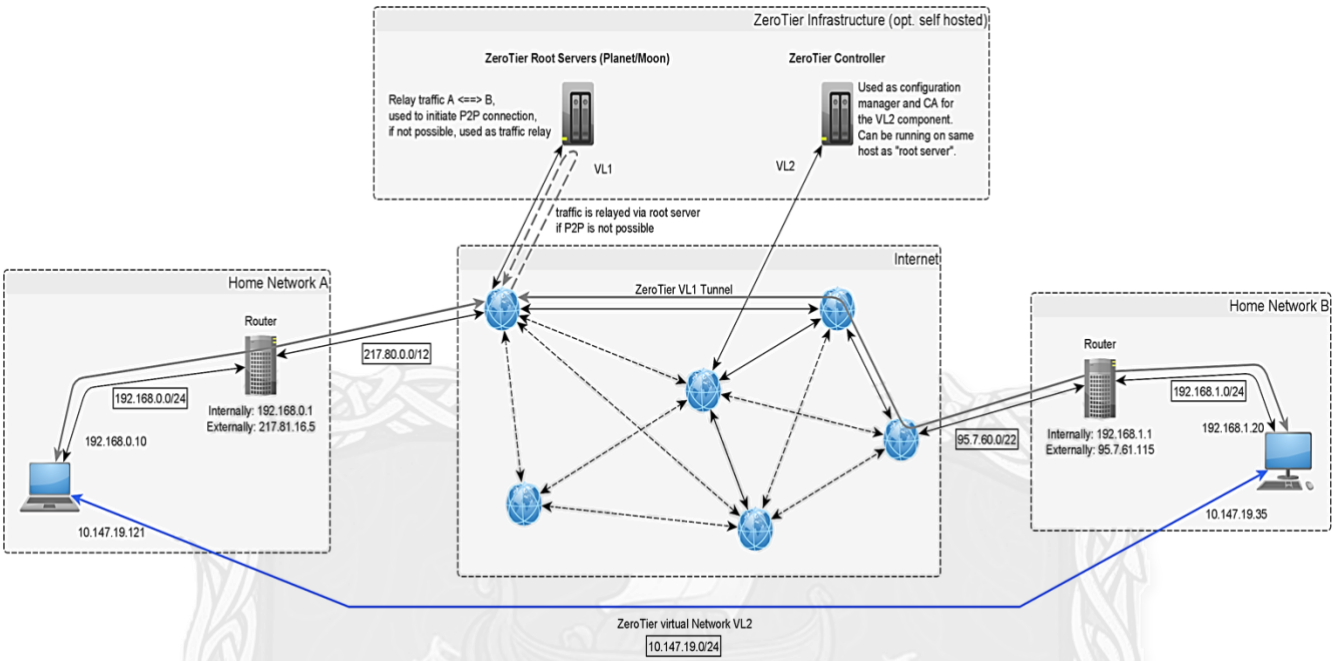
Otra información necesaria para su realización

En esta sección vamos a tratar de explicarte cómo aprender a usar **un servicio VPN P2P llamado *ZeroTier***, que permitiría acceder a cualquier dispositivo desde cualquier parte de una manera descentralizada (sin depender de un servidor central como *ProtonVPN*), segura y más sencilla de instalar que una VPN propia.

🔍 ***Esta VPN P2P es ideal para implementar teletrabajo en nuestra empresa futura o conectarnos a nuestra casa teniendo un control completo de todo el servicio que lo proporciona y, normalmente, de manera gratuita porque no necesitarás mucha infraestructura en estos casos***

Servicios **VPN punto a punto** como ***TailScale*** o ***ZeroTier*** son los que permiten conectar fácilmente muchos dispositivos que se vean directamente (como en una LAN) sin importar dónde están en el mundo. Gracias a ellos podemos instalar su *software* **en cada uno de nuestros dispositivos que deseemos usar desde sitios remotos** y, de esta forma, en lugar de depender de un servidor central, formar una red de dispositivos que

se pueden ver entre ellos como si estuvieran en una red local, independientemente del sitio del mundo en el que estén. La siguiente imagen (https://blog.rico-j.de/zerotier-one/) muestra cómo funcionan esta clase de VPNs



> 🔍 ***Esta es la principal ventaja de estas tecnologías que, al eliminar el punto central, nos quita problemas de disponibilidad y de rendimiento. También nos liberamos del problema de hacer visible el servicio hacia el exterior, porque habitualmente el cliente se encarga de hacer toda la gestión por nosotros***

Por tanto, tendríamos una forma sencilla de **compartir recursos que tenemos de una manera segura**, minimizando el riesgo de que alguien no autorizado sea capaz de acceder a ellos.

Instalación de *ZeroTier*

Para instalar *ZeroTier* lo primero que debemos hacer es crearnos una cuenta gratuita aquí: https://my.zerotier.com/login. El registro **exige una cuenta de correo electrónico real** porque requiere verificarla. Una vez registrados, veremos la pantalla de bienvenida. En ella veremos el botón "***Create A Network***" para crear una VPN, pero ya tendremos una creada de antemano que nos sirve perfectamente para hacer pruebas. Podemos crear varias VPN si queremos. *¿Para qué crees que serviría esto?*

> 🔍 ***Cada red de ZeroTier tiene un identificador único de 48bits (que será necesario usar luego para que los dispositivos se vinculen a la red creada. Además, se le asigna un nombre provisional aleatorio, pero que es conveniente cambiar por otro que tenga sentido para nosotros.***

Es importante dejar la red como "PRIVATE" para que cualquier dispositivo que se una a la misma **tengas que aprobarlo** previamente. Las redes públicas no requieren aprobación para conectarse, y son para otros escenarios. En la sección ***Advanced*** hay más opciones, pero **con la configuración por defecto debería funcionar sin problemas**.

> 🔍 ***ZeroTier selecciona automáticamente un rango de IPs para la VPN que se va a crear. Fíjate que son direcciones IP privadas, como las que se mencionaron en el tema 1 de teoría***

En la sección ***Members*** es donde se autorizan los dispositivos que queréis tener activos en la VPN. También veréis los datos de la última vez que se conectó un dispositivo, entre otra información, lo cual es muy útil para detectar posibles intrusos o usos no autorizados. No obstante, inicialmente aparecerá una red sin dispositivos añadidos, como es lógico.

La sección "***Flow Rules***" **nos permite configurar un *firewall* y sus reglas**, lo que nos deja decidir el tipo de tráfico que queremos admitir o no en la red (protocolos, puertos, etc.). Para este ejercicio podemos dejar la configuración por defecto.

Añadir dispositivos

Para añadir dispositivos a la red recién creada **tenemos que instalar el *software* que controla la VPN en cada uno de ellos**. Tenemos clientes para los dispositivos que mencionamos antes aquí: https://www.zerotier.com/download/. Para distribuciones *Linux* tenemos que instalar el *software* que haga de cliente usando alguno de los métodos indicados en la web de *ZeroTier*. Todo el proceso de instalación y de añadir cada cliente a la red **se realiza bajo línea de comandos**.

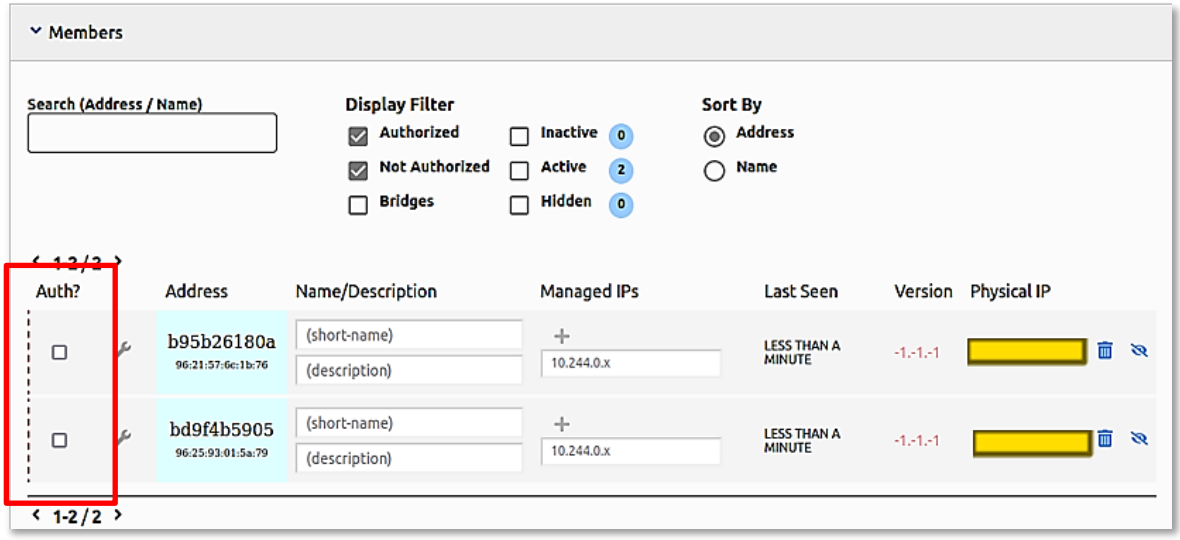
🔍 ***También hay una app de Android oficial, si queremos usar como cliente nuestro teléfono: https://play.google.com/store/apps/details?id=com.zerotier.one***

Una vez instalado el cliente, desde *Ubuntu*, en cada uno de los dispositivos que vayan a formar parte de la red debemos hacer previamente dos cosas :

- Asegurarnos de que está instalado ***curl***: `sudo apt-get install curl`
- Ejecutar `sudo zerotier-one -d` para arrancar el servicio en cada nodo que se vaya a añadir a la red

Ahora ya podemos añadir cada uno de los dispositivos que forman parte de la red ***con*** `sudo zerotier-cli join <el ID de red completo que nos salió en la web de ZeroTier>`. El resto de las opciones se pueden dejar como están. Como la red que creamos es privada y hemos usado el dispositivo para solicitar ser añadidos a ella, en la web de administración de *ZeroTier* nos aparecerán los **dispositivos que se han unido a nuestra VPN**. Es ahora cuando **debemos aceptar** que se conecten para poder seguir trabajando.

🔍***¿Cómo es posible que se hayan añadido los dispositivos sin más? Porque conocían nuestro ID de red. Si alguien lo averigua (no deberíamos revelárselo a nadie ajeno) podrá tratar de conectar un dispositivo, pero ese dispositivo solo podrá ver a los otros en la red si activamos su casilla de autorización, como veremos enseguida***



Si hacemos clic en la columna "***Auth?***", daremos acceso a los dispositivos que queramos y ya los podemos usar en la red, pudiéndose ver entre ellos de la misma forma que si estuvieran en una red local.

🔍 ***Conviene no obstante cambiarles el nombre para poder identificarlos fácilmente cuando la red crezca de tamaño.***

Al hacerlo, desde ese momento *ZeroTier* **asignará una IP** del rango privado que seleccionamos antes al dispositivo cliente y entonces podrá usarla. Para añadir más dispositivos a la VPN, y así poder establecer una conexión entre todos y probar que realmente una VPN punto a punto cumple con su labor, hay que repetir el proceso de instalación del cliente visto en cada dispositivo.

Para hacer las pruebas se puede probar a hacer un `ping` entre dispositivos usando la IP asignada por *ZeroTier* a cada uno de ellos, o bien intentar acceder a un servicio que un miembro de la red ofrezca a los otros (por ejemplo, una web, el escritorio remoto...). Si funciona, lo habremos hecho todo correctamente.

🔍 ***ZeroTier puede funcionar dentro de máquinas virtuales que trabajen con una conexión NAT para salir Internet. Esto quiere decir que desde cualquier parte del mundo puedes conectarte a dentro de una MV tuya, por lo que no solo estás asegurando la conexión, sino que, además, si tu dispositivo está comprometido y el atacante puede usarlo para conectarse a tu VPN P2P, sus acciones estarán contenidas dentro de la MV correspondiente, limitando mucho su capacidad de causar problemas***

Insignias y Autoevaluación

NOTA: Tienes una versión de esta tabla de insignias en formato editable disponible en el *Campus Virtual*. Puedes usar este archivo para crear un documento en el formato que desees y tomar notas extendidas de tus actividades para crear un log de lo que has hecho. Recuerda que el material que elabores se puede llevar a los exámenes de laboratorio.

Nivel de Insignia	Desbloqueado cuando	¿Desbloqueado?
	Sabes cómo permitir o denegar servicios/puertos de un *firewall*	
	Puedes permitir o denegar conexiones desde direcciones IP individuales o redes	
	Puedes responder a esta pregunta: *¿Qué tipo de ataques podría mitigar potencialmente que un firewall ponga un límite de peticiones/tiempo a un servicio?*	
	Puedes responder a esta pregunta: *¿Cuál crees que es el propósito de registrar / auditar intentos de conexiones explícitamente prohibidas en el firewall?*	
	Puedes responder a estas preguntas: *¿Por qué crees que es útil bloquear a las máquinas que realizan escaneos?* ***PISTA****: ¿Escaneas máquinas como parte de tu rutina normal?*	
	Entiendes por qué el bloqueo de IPs vía DNS es útil. Puede responder a esta pregunta: *¿Cuál es la ventaja de usar PiHole para bloquear IPs en lugar de un complemento de un navegador que bloquee IPs maliciosas?*	
	Puedes responder a estas preguntas: *¿Crees que* `fail2ban` *reemplaza a un firewall? ¿o más bien lo complementa?*	
	Puedes responder a esta pregunta: *¿Qué tipo de ataques previene* `fail2ban?`	
	Puedes responder a esta pregunta: *¿Crees que un bloqueo temporal es suficiente para prevenir un intento de ataque DoS?*	
	Puedes responder a estas preguntas: *¿Por qué crees que dejar los puertos abiertos a propósito puede ser una medida útil de detección de escaneos? Por lógica, ¿qué puertos dejarías abiertos de esta manera?*	
	Puedes responder a estas preguntas: *¿Qué pasa cuando una máquina está bloqueada por* `portsentry`*? ¿La máquina bloqueada puede ponerse en contacto con la que bloquea usando cualquier servicio o no?*	
	Puedes responder a estas preguntas: *¿Afectan los bloqueos de* `portsentry` *a la velocidad de escaneo? Si la respuesta es sí, ¿crees que esto también se puede utilizar como medida de seguridad?*	
	Entiendes para qué sirve un NIDS y el tipo de cosas que detecta respecto al tráfico de red que pasa por él	

	Puedes responder a esta pregunta: *¿Cuántas cosas crees que puedes hacer para proteger las* conexiones `ssh` *desde el punto de vista de la red?* (piensa en combinar técnicas que vimos aquí y laboratorios anteriores)	
	Puedes protegerte de los intentos de ataque a `ssh` (y ataques similares a otros servicios que hacen) con *jails* predefinidas de `fail2ban`.	
	Puedes habilitar una "trampa de escaneo de puertos" en una máquina sin entrar en conflicto con el comportamiento normal del *firewall*. Puedes responder a esta pregunta: *si puedes recopilar las IP bloqueadas, ¿qué crearías con ellas?*	
	Puedes construir una red de dispositivos tuyos con una VPN P2P con *ZeroTier* y gestionarlos de forma segura, así como distinguir estas VPN de las tradicionales	