Raúl Fernández del Blanco
3ºGEI

PRÁCTICA 1 LSI

a) Configure su máquina virtual de laboratorio con los datos proporcionados por el profesor. Analice los ficheros básicos de configuración (interfaces, hosts, resolv.conf, nsswitch.conf, sources.list, etc.)

El primer paso para configurar la máquina es cambiarle la contraseña, tanto al usuario como al root. Esto se puede conseguir con los siguientes comandos:

Cambio de la contraseña de usuario: `lsi@debian:~$ sudo passwd lsi`

Cambio de la contraseña de root: `lsi@debian:~$ sudo passwd`

Luego, seguimos desactivando los servicios que se encargan de detectar automáticamente los recursos de una red local y conectarse a ella.

Para conocer los distintos procesos que hay en la máquina tenemos el siguiente comando:

```
lsi@debian:~$ ps -eaf
```

Además, para ver los distintos servicios que están corriendo en nuestra máquina tenemos el comando:

```
lsi@debian:~$ systemctl --all
```

Es momento de desactivar los servicios:

```
lsi@debian:~$ systemctl stop avahi-daemon.socket
lsi@debian:~$ systemctl stop avahi-daemon.service
lsi@debian:~$ systemctl stop NetworkManager.service
```

```
lsi@debian:~$ systemctl disable avahi-daemon.socket
lsi@debian:~$ systemctl disable avahi-daemon.service
lsi@debian:~$ systemctl disable NetworkManager.service
```

Ahora pasamos a los ficheros básicos de configuración. En primer lugar, tenemos el fichero /etc/network/interfaces (en el que se configura la red de nuestros servidores), en el que modificaremos/añadiremos lo siguiente:

```
auto lo ens33 ens34
iface lo inet loopback
iface ens33 inet static
address 10.11.49.116
netmask 255.255.254.0
broadcast 10.11.49.255
network 10.11.48.0
gateway 10.11.48.1
iface ens34 inet static
address 10.11.51.116
netmask 255.255.254.0
broadcast 10.11.51.255
network 10.11.50.0
```

En este fichero podemos comprobar varias palabras clave:

- Auto: indica que el puerto se levantará después de un reset.
- Iface: sirve para indicar el interfaz sobre el que se está trabajando.
- Inet: identifica cómo va a obtener el direccionamiento el puerto:
    - Static: direccionamiento a mano.
    - Dhcp: direccionamiento por dhcp.

Otro de los ficheros básicos de configuración es el fichero /etc/resolv.conf. En este fichero nos encontramos las direcciones de los servidores DNS de nuestra máquina.

El fichero /etc/apt/sources.list contiene las direcciones web de los diferentes repositorios del sistema operativo. A partir de este archivo podremos realizar actualizaciones del sistema operativo (Ej: actualizar debian 10 a debian 11).

El fichero /etc/hosts se utiliza para obtener una relación entre un nombre de máquina y una dirección IP, de forma que un usuario no tenga que recordar direcciones sino nombres de hosts. Se suelen incluir las direcciones y nombres de todos los equipos conectados a la red local para que la comunicación local no se tenga que realizar a través de DNS.

Por último, el fichero /etc/nsswitch.conf, también llamado Name Service Switch Configuration file, tiene como objetivo especificar de donde obtienen la información ciertos valores especiales relacionadas con la librería de C y, en caso de obtener la información de varios medios, decidir en qué orden consultar éstos con el fin de saber qué fuente tiene más prioridad. Dentro de este archivo nos podemos encontrar con varias referencias que nos pueden facilitar la vida:

- Db: busca en una librería toda la información relacionada con la categoría buscada.
- Files: la referencia más útil de todas. Cada vez que busquemos en files, nsswitch buscará en /etc un fichero con el mismo nombre.

Una vez modificados los archivos de configuración, tenemos que hacer el siguiente comando, que reiniciará el servicio de red:

```
lsi@debian:~$ systemctl restart networking.service
```

b) ¿Qué distro y versión tiene la máquina inicialmente entregada? Actualice su máquina a la última versión estable disponible.

Primero, tenemos que comprobar la versión de nuestra máquina. En un principio, dispondrá de una versión muy antigua del sistema operativo Debian, en concreto Debian 10 con repositorios buster. Para actualizar la máquina, primero deberemos realizar una copia de seguridad como medida de prevención. Luego, tendremos que actualizar debian 10 a su última versión y por último tendremos que actualizar los repositorios y el sistema. Para que los cambios se hagan efectivos debemos hacer un reboot de la máquina y comprobar la última versión de debian.

```
lsi@debian:~$ lsb_release -a
lsi@debian:~$ cp /etc/apt/sources.list /etc/apt/sources.list.backup
lsi@debian:~$ apt update
lsi@debian:~$ apt full-upgrade
lsi@debian:~$ sed -i ‘s/búster/bullseye/g’ /etc/apt/sources.list
lsi@debian:~$ apt update
lsi@debian:~$ apt full-upgrade
lsi@debian:~$ reboot
lsi@debian:~$ lsb_release -a
```

De esta forma, habremos actualizado debian de una forma fácil y segura.

c) Identifique la secuencia completa de arranque de una máquina basada en la distribución de referencia (desde la pulsación del botón de arranque hasta la pantalla de login). ¿Qué target por defecto tiene su máquina? ¿Cómo podría cambiar el target de arranque? ¿Qué targets tiene su sistema y en qué estado se encuentran? ¿Y los services? Obtenga la relación de servicios de su sistema y su estado. ¿Qué otro tipo de unidades existen?

En cuanto a la secuencia de arranque, lo primero que tenemos que saber es que el fichero de configuración del gestor de arranque es el /etc/default/grub. Además de esto, también es de gran interés saber que el siguiente comando nos permite ver los logs del boot actual de la máquina:

```
lsi@debian:~$ journalctl -b
```

En cuanto a la secuencia de arranque, podríamos describirla en los siguientes pasos:

1. El hardware carga la BIOS (que se almacena en una memoria no volátil). Este hace comprobaciones de hardware e inicializa el hardware necesario para arrancar. Después de inicializarlo, busca el sector de arranque, en nuestro caso MBR.
2. El MBR se encarga de cargar y ejecutar el gestor de arranque, en nuestro caso GRUB.
3. El GRUB es el programa de carga de arranque para sistemas operativos tipo Linux. Este se encarga de iniciar y cargar el Kernel, es decir, se inicializa el núcleo del sistema y se monta la partición raíz.
4. Se ejecuta el proceso init, el padre de todos los procesos y fichero raíz en RAM. Init configura todos los servicios y estructuras que no sean del Sistema Operativo y entrega el control a systemd.
5. Systemd inicia grupos de procesos y servicios en paralelo organizados en targets.

Un **target** se utiliza para la agrupación lógica de unidades, es decir, referencia a otras unidades que pueden ser controladas (ejecutadas/paradas/destruidas) conjuntamente.

Para conocer nuestro target por defecto, tenemos que ejecutar el siguiente comando:

```
lsi@debian:~$ systemctl get-default
```

En un principio, nuestra máquina tendrá por defecto el target **graphical.target.** Sin embargo, como no vamos a utilizar una interfaz gráfica en nuestra máquina, lo óptimo para la máquina sería cambiar de target para que no se ejecute esta interfaz gráfica. Por ello, tendremos que cambiar nuestro target de arranque al **multi-user.target**, utilizando el siguiente comando:

```
lsi@debian:~$ systemctl set-default multi-user.target
```

Para conocer los distintos targets de nuestro sistema y saber en qué estado se encuentran, existe un comando que nos podía ser de utilidad:

```
lsi@debian:~$ systemctl list-units --type=target -all
```

Y para conocer los servicios y sus respectivos estados tenemos el comando:

```
lsi@debian:~$ systemctl list-units --type=service -all
```

En ambos casos, sus estados pueden ser:

- Enabled: el servicio está habilitado/disponible.
- Disabled: el servicio está deshabilitado.
- Static: servicios que únicamente se usarán en el caso que otro servicio lo precise. Pueden estar activos o inactivos, pero siempre están disponibles para cuando se necesite.
- Masked: el servicio está completamente deshabilitado y no se puede iniciar de ningún modo sin previamente desenmascararlo.

Además de los targets y los services, existen otro tipo de unidades como:

- Sockets: esta unidad encapsula un socket en el sistema de archivos o en internet. Cada socket tiene una unidad de servicio correspondiente.
- Device: esta unidad encapsula un dispositivo en el árbol de dispositivos de Linux.
- Mount: esta unidad encapsula un punto de montaje en la jerarquía del sistema de archivos.
- Timer: son servicios que ejecutan funciones con cierta periodicidad.
- Path, slice, scope, swap, automount.

d) Determine los tiempos aproximados de botado de su kernel y del userspace. Obtenga la relación de los tiempos de ejecución de los services de su sistema.

Para obtener los tiempos de bootado tanto del kernel como del userspace, simplemente tenemos que ejecutar el siguiente comando:

```
lsi@debian:~$ systemd-analyze
```

Por último, para obtener los tiempos de ejecución de cada uno de los servicios del sistema, simplemente tenemos que realizar el siguiente comando:

```
lsi@debian:~$ systemd-analyze blame
```

e) Investigue si alguno de los servicios del sistema falla. Pruebe algunas de las opciones del sistema de registro journald. Obtenga toda la información journald referente al proceso de botado de la máquina. ¿Qué hace el systemd-timesyncd?

Para obtener los servicios del sistema que fallan, se pueden ejecutar alguno de los siguientes comandos:

```
lsi@debian:~$ journalctl -xe | grep fail
lsi@debian:~$ systemctl list-unit-files --state=failed
```

En cuanto a las distintas opciones del registro journald:

- Journalctl: muestra todos los logs del sistema.
- Journalctl -f: muestra los mensajes que se van generando en tiempo real.
- Journalctl -r: permite revertir la salida para que las entradas más recientes se muestren primero.
- **Journalctl -u <servicio>**: investiga sobre cómo y cuándo se ha usado ese servicio.
- **Journalctl -b**: muestra los logs del botado de la máquina.
- Journalctl --list-boots: esta opción sirve para ver la lista de todos los boots que existen en el sistema.
- Journalctl –disk-usage: permite ver el espacio que está siendo ocupado por los diferentes logs.

El comando systemd-timesyncd es un daemon que sirve para sincronizar el reloj del sistema a través de la red, utilizando el protocolo SNTP.

f) Identifique y cambie los principales parámetros de su segunda interface de red (ens34). Configure una segunda interface lógico. Al terminar, déjelo como estaba.

En primer lugar, un comando muy interesante para comprobar la configuración actual de nuestras interfaces de red es:

```
lsi@debian:~$ ip a s
```

El parámetro **a** nos permite ver todos los interfaces de red y el parámetro **s** nos permite ver las estadísticas de estos interfaces.

Además de ver esto, también podemos ver la configuración de estas interfaces en el fichero /etc/network/interfaces. Para configurar una segunda interfaz lógica, primero tenemos que crearla:

```
lsi@debian:~$ ifconfig ens34:0 10.11.51.253 netmask 255.255.254.0
```

De esta forma, creamos la interfaz lógica 0 en la interfaz física ens34 de nuestra máquina. Podemos comprobar que se ha configurado el interfaz lógico si ejecutamos el primer comando del ejercicio y para comprobar directamente qué subredes están asociadas a la interfaz podemos usar el siguiente comando:

```
lsi@debian:~$ ip a ls ens34 | grep inet
```

Una vez configurada esta nueva interfaz lógica, podemos comprobar que se ha creado correctamente haciendo un ping sobre la nueva dirección.

En caso de querer eliminar de nuestra configuración esta interfaz, simplemente tenemos que ejecutar el siguiente comando:

```
lsi@debian:~$ ip a del 10.11.51.253/23 dev ens34:0
```

g) ¿Qué rutas (routing) están definidas en su sistema? Incluya una nueva ruta estática a una determinada red.

Para determinar la tabla de enrutamiento de nuestro sistema podemos ejecutar uno de los siguientes comandos por consola:

```
lsi@debian:~$ route -n
lsi@debian:~$ netstat -r
```

Como podemos observar, la conexión hacia el exterior se realiza a través del interfaz ens33, mientras que el interfaz ens34 podremos utilizarlo como red auxiliar para realizar cualquier tipo de pruebas sin tener miedo a cargarnos la red y no poder volver a conectarnos a la máquina.

Para añadir una nueva ruta a nuestra tabla de enrutamiento (en nuestro caso vamos a añadir la red 10.11.52.0), vamos a usar el siguiente comando (IMPORTANTE: meter la máscara de red y el Gateway en la misma sentencia porque si no te tira la máquina):

```
lsi@debian:~$ route add -net 10.11.52.0 netmask 255.255.254.0 gw 10.11.50.1
```

Gracias a este comando, añadimos temporalmente (en caso de hacer un reboot o reiniciar el servicio de red, esta nueva ruta se borraría) esta ruta a nuestra máquina.

En caso de querer eliminar esta nueva ruta, tenemos el siguiente comando:

```
lsi@debian:~$ route del -net 10.11.52.0 netmask 255.255.254.0 gw 10.11.50.1
```

h) En el apartado d) se ha familiarizado con los services que corren en su sistema. ¿Son necesarios todos ellos? Si identifica servicios no necesarios, proceda adecuadamente. Una limpieza no le vendrá mal a su equipo, tanto desde el punto de vista de la seguridad, como del rendimiento.

Uno de los motivos por los que nos viene bien deshabilitar servicios de nuestra máquina es porque nos consume muchos recursos a la hora de botado de la máquina y tarda mucho tiempo, por lo que iremos eliminando aquellos servicios que consuman más y no sean necesarios en nuestro sistema (comando: systemd-analyze blame). Además, en este apartado solamente nos piden eliminar los servicios que estén en estado **enabled,** por lo que simplemente buscaremos aquellos que sigan esta norma. Gracias al siguiente comando podremos saber cuáles son:

```
lsi@debian:~$ systemctl list-unit-files --state=enabled
```

En mi máquina he decidido eliminar los siguientes servicios:

- Bluetooth.service: no necesitamos bluetooth en nuestra máquina.
- Switcheroo-control.service: verifica la disponibilidad de GPU dual.
- ModemManager.service: controla conexiones 3G y 4G. En nuestro caso no contamos con estas conexiones.
- Wpa_supplicant.service: controla el roaming y la autenticación IEEE del controlador wland. Permite la conexión a redes cifradas con WPA, en nuestro caso no se necesita.
- cups-browsed.service: ayuda al descubrimiento de impresoras remotas en la red.
- cups.service/cups.socket: este servicio permite configurar un equipo como servidor de impresión en Linux, además de poder instalar impresoras y compartirlas.
- tpm2-abrmd.service: es un servicio que se comunica con el chip TPM, en nuestro caso las maquinas carecen de él.
- Accounts-daemon.service: es útil deshabilitarlo, ya que en caso de que sea necesario este servicio, se inicia de forma automática por DBUS (mecanismo de comunicación entre procesos).
- Console-setup.service: permite realizar personalizaciones en la consola. En nuestro caso, no necesitamos hacer ninguna modificación.
- Cron: Es parecido a anacron, pero con menos cosas.
- e2scrub_reap.service: intenta verificar todos los metadatos en un sistema de archivos lógico LVM (busca errores). Lo desactivamos, ya que está entre graphic.target y multi-user.target, por lo que nosotros nunca lo usaremos.
- Servicios de VMWare como: vgauth.service u open-vm-tools.service.
- Udisks2.service: proporciona interfaces para enumerar y realizar operaciones en discos y dispositivos de almacenamiento. Udisks solo está involucrado indirectamente en qué dispositivos y objetos se muestran en la interfaz de usuario.

Los servicios que permanecen activos son:

- Anacron.service: Similar al servicio cron, el servicio anacron ejecuta aplicaciones o scripts en fechas y horas específicas. A diferencia del servicio cron, anacron no perderá la ejecución de un trabajo programado, incluso si el sistema está apagado. La actividad se realizará la próxima vez que el sistema esté disponible. Esto convierte a anacron en la opción preferida para iniciar tareas esenciales de administración del sistema, como la copia de seguridad o la recuperación de espacio en disco.
- Apparmor.service: proporciona seguridad de control de acceso obligatorio.

- Systemd-pstore.service: un sistema de archivos de almacenamiento persistente, pstore, que puede almacenar registros de errores cuando el kernel muere (o se reinicia o se apaga). A su vez, se puede hacer referencia a estos registros para depurar problemas del kernel
- unattended-upgrades.service: su propósito es mantener a la máquina con las últimas actualizaciones de seguridad automáticamente.
- Getty@.service: Su propósito es proteger el sistema del acceso no autorizado. En general, cada proceso getty se inicia mediante systemd y gestiona una sola línea de terminal.
- Keyboard-setup.service: no hay mucha información a cerca de este servicio. Como nosotros utilizamos constantemente el teclado, he preferido no desactivar este servicio.

i) Diseñe y configure un pequeño "script" y defina la correspondiente unidad de tipo service para que se ejecute en el proceso de botado de su máquina.

Antes de nada, tenemos que crear un script .sh. En nuestro caso, vamos a hacer un script en el que cada vez que se haga un reboot del sistema, escriba en un archivo la memoria que tiene la máquina en ese momento y nos diga si es preocupante la cantidad de memoria o no. Este archivo se encuentra en la ruta /usr/local/bin y tiene permisos de ejecución.

Una vez creado el script, nos dirigimos a la carpeta /lib/systemd/system, en la que se encuentran todos los servicios del sistema y creamos un servicio nuevo en el que introducimos lo siguiente:

```
[Unit]
Description=Script para la practica 1 de lsi
After=network.target

[Service]
Type=forking
ExecStart=/usr/local/bin/mi-script.sh

[Install]
WantedBy=multi-user.target
```

Los parámetros de este archivo son los siguientes:

- Description: indicas una descripción del servicio.
- After: Indicas qué servicios necesitas que se carguen antes que el tuyo. En nuestro caso es network.target, ya que queremos que queremos garantizar que el servicio de red haya comenzado
- ExecStart: especifica la ruta en la que se encuentra el script.
- WantedBy: indica el target que solicita la ejecución del servicio. En debian hay un runlevel que son los niveles de ejecución y que van del 0 al 6, siendo 0 el apagado y 6 el reinicio. Cuanto más alto es, mas complejas son las cosas que hace el sistema. El runlevel 5 es el graphical.target. Como este no nos sirve, cogemos el más bajo que nos permite tener multiusuario y conexión de red, en nuestro caso, multi-user.target.

Una vez guardado el archivo, en nuestro caso con el nombre p1-servicio.service hay que hacer lo siguiente:

```
lsi@debian:~$ systemctl daemon-reload
lsi@debian:~$ systemctl enable p1-servicio.service
```

De esta forma, activamos el servicio para que se inicie con cada inicio del sistema. Para comprobar que el servicio funciona correctamente, solo tenemos que hacer un reboot y comprobar que se ha escrito algo en el fichero que hemos indicado en el script.

j) Identifique las conexiones de red abiertas a y desde su equipo.

Para identificar las conexiones de red abiertas tenemos que utilizar una herramienta llamada **netstat** que entrega estadísticas básicas sobre la totalidad de las actividades de red. También puede entregar información acerca de los puertos y direcciones a través de los cuales se ejecutan las conexiones TCP y UDP, al igual que los puertos abiertos para solicitudes.

El comando más utilizado para ver todas las conexiones de red activas y puertos abiertos es:

```
lsi@debian:~$ netstat -ano
```

Con este comando tenemos más argumentos que nos pueden proporcionar diferente información:

- Netstat -p <protocolo>: lista las conexiones para el protocolo especificado.
- Netstat -t: lista todos los puertos TCP.
- Netstat -u: lista todos los puertos UDP.
- Netstat -a: muestra todas las conexiones activas y los puertos abiertos.
- Netstat -e: muestra estadísticas.
- Netstat -n: visualización numérica de las direcciones y números de puerto.
- Netstat -q: lista todas las conexiones y todos los puertos TCP.
- Netstat -l: lista solo los puertos que están escuchando.

Además de este comando, tenemos varios más que nos pueden ser de utilidad:

- Lsof: lista los archivos abiertos por diferentes procesos en ejecución. Para nuestro caso, vamos a querer detalles de las conexiones de red abiertas (i).

  ```
  lsi@debian:~$ lsof -ni
  ```

- Ss: Es parecida a netstat. Con el siguiente comando podemos ver información sobre todos los puertos abiertos:

  ```
  lsi@debian:~$ ss -a
  ```

k) Nuestro sistema es el encargado de gestionar la CPU, memoria, red, etc., como soporte a los datos y procesos. Monitorice en "tiempo real" la información relevante de los procesos del sistema y los recursos consumidos. Monitorice en "tiempo real" las conexiones de su sistema.

Para realizar la tarea de monitorización en tiempo real de nuestro sistema existen variedad de opciones, las más utilizadas son los comandos:

- top: este comando te permite ver las tareas del sistema que se ejecutan en tiempo real.
- htop: es una herramienta muy parecida a la anteriormente explicada con la diferencia que es más interactiva y visual y resulta bastante más cómoda a la hora de visualizar procesos y cerrar alguno que sea de interés.

l) Un primer nivel de filtrado de servicios los constituyen los tcp-wrappers. Configure el tcp-wrapper de su sistema (basado en los ficheros hosts.allow y hosts.deny) para permitir conexiones SSH a un determinado conjunto de IPs y denegar al resto. ¿Qué política general de filtrado ha aplicado? ¿Es lo mismo el tcp-wrapper que un firewall? Procure en este proceso no perder conectividad con su máquina. No se olvide que trabaja contra ella en remoto por SSH.

En un primer lugar, editamos el archivo /etc/hosts.allow, en el que pondremos lo siguiente:

```
#MI_IP
sshd: 127.0.0.1 :spawn /bin/echo "($(date)|MILOCAL) from %a service %d
>>/var/log/conexion_ssh.txt
sshd: 10.11.49.116 :spawn /bin/echo "($(date)|MIIP) from %a service %d
>>/var/log/conexion_ssh.txt
sshd: 10.11.51.116 :spawn /bin/echo "($(date)|MIIP34) from %a service %d
>>/var/log/conexion_ssh.txt

#IP_COMPI
sshd: 10.11.49.48 :spawn /bin/echo "($(date)|COMPIIP) from %a service %d
>>/var/log/conexion_ssh.txt
sshd: 10.11.51.48 :spawn /bin/echo "($(date)|COMPIIP34) from %a service %d
>>/var/log/conexion_ssh.txt

#IP_VPN
sshd: 10.30.:spawn /bin/echo "($(date)|VPN) from %a service %d
>>/var/log/conexion_ssh.txt

#IP_EDUROAM
sshd: 10.20.32.0/21 :spawn /bin/echo "($(date)|VPN) from %a service %d
>>/var/log/conexion_ssh.txt
```

Una vez configurado el fichero hosts.allow, configuramos el deny, en el que negaremos el resto de las entradas, permitiendo así solo las conexiones configuradas anteriormente. En el fichero hosts.deny introducimos lo siguiente:

```
ALL:ALL : twist /bin/echo "%h has been banned from this server!"
```

La política de bloqueo utilizada es una muy restrictiva ya que bloqueamos cualquier entrada diferente a las que habilitamos expresamente en el fichero host.allow.

Los TCP Wrappers suelen utilizarse para filtrar direcciones ip y hostnames. Un firewall es un dispositivo de hardware o un software que nos permite gestionar y filtrar la totalidad de tráfico entrante y saliente que hay entre 2 redes u ordenadores de una misma red. Los wrappers también funcionan a nivel de app y los firewalls no.

Para configurar los TCP Wrappers hemos tenido que usar los siguientes comandos y extensiones:

- Spawn: lanza un proceso hijo con el comando especificado después.
- Twist: Reemplaza el servicio solicitado con el comando especificado. Se utiliza para colocar trampas para intrusos.
- %a: IP del cliente. %h: nombre de la máquina del servidor. %d: nombre del proceso.

m) Existen múltiples paquetes para la gestión de logs (syslog, syslog-ng, rsyslog). Utilizando el rsyslog pruebe su sistema de log local.

Rsyslog se encarga de implementar el protocolo de syslog básico, lo extiende con filtrado basado en un contenido dado, con capacidades de filtrado enriquecido, opciones de configuración flexibles y agrega características como uso de TCP para el transporte.

Para probar el funcionamiento del log local de nuestro sistema basta con generar un log a algún recurso dado y verificar si efectivamente esta archivado en el log correspondiente:

```
lsi@debian:~$ logger -p mail.err “Prueba de error”
```

Para verificar el correcto funcionamiento ingresamos en el usuario root y visualizamos el fichero /var/log/mail.err.

n) Configure IPv6 6to4 y pruebe ping6 y SSH sobre dicho protocolo. ¿Qué hace su tcp-wrapper en las conexiones SSH en IPv6? Modifique su tcp-wapper siguiendo el criterio del apartado.

Añadimos al fichero interfaces la nueva interfaz 6to4:

```
auto lo ens33 ens34 6to4
|
|
|
iface 6to4 inet6 v4tunnel
address 2002:a0b:3174::1
netmask 16
endpoint any
local 10.11.49.116
```

Y luego modificamos el fichero hosts.allow:

```
#MI_IP
sshd: [2002:a0b:3174::1] :spawn /bin/echo "($(date)|MIPV6) from %a
service %d >>/var/log/conexion_ssh.txt
#MI_COMPI
sshd: [2002:a0b:3130::1] :spawn /bin/echo "($(date)|SUIPV6) from %a
service %d >>/var/log/conexion_ssh.txt
```

Para que mi tcp-wrapper pueda funcionar en IPv6 tenemos que añadirle las líneas escritas anteriormente en el host.allow para permitir las conexiones. De la otra manera denegará todo tipo de conexiones ipv6.

h) ¿Necesita IPv6? ¿Cómo se deshabilita IPv6 en su equipo

En la actualidad se sigue utilizando el protocolo IPv4 como protocolo de nivel de red principal, es por esto por lo que en nuestro caso y en general no es necesario tener el protocolo IPv6 habilitado, para ello editaremos el archivo /etc/sysctl.conf y le añadiremos al final de este las siguientes líneas:

```
net.ipv6.conf.all.disable_ipv6 = 1
net.ipv6.conf.default.disable_ipv6 = 1
net.ipv6.conf.lo.disable_ipv6 = 1
```

PARTE GRUPAL:

a) En colaboración con otro alumno de prácticas, configure un servidor y un cliente NTP.

En el servidor configuramos:

```
#server ntp.your-provider.example
server 127.127.1.1 minpoll 4 prefer
fudge 127.127.1.1 stratum 10
restrict default ignore
restrict 10.11.49.116 mask 255.255.255.255 noquery nopeer
restrict 127.127.1.1 mask 255.255.255.255 noserve nomodify

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1 mask 255.255.255.255 noserve nomodify
restrict ::1

# Needed for adding pool entries
restrict source notrap nomodify noquery
```

En el cliente configuramos:

```
server 10.11.49.48 minpoll 4 prefer

# Local users may interrogate the ntp server more closely.
restrict 127.0.0.1
restrict ::1

# Needed for adding pool entries
restrict source notrap nomodify noquery

#Acceso desde la misma máquina, para supervisión
restrict 127.0.0.1 mask 255.255.255.255

#Acceso al servidor
restrict 10.11.49.48
```

b) Cruzando los dos equipos anteriores, configure con rsyslog un servidor y un cliente de logs.

En el servidor configuramos:

```
#Provides TCP syslog reception
module(load=”imtcp”)
input(type=”imtcp” port=”514”)
$AllowedSender TCP, 127.0.0.1, 10.11.49.48

##RULES##
$template remote,"/var/log/%FROMHOST-IP%/%PROGRAMNAME%.log"
:inputname, isequal, "imtcp" ?remote
&stop
```

En el cliente configuramos:

```
*.* action(type="omfwd"
        queue.type="LinkedList"
        queue.filename="cola_logs"
        queue.maxFileSize="1m"
        action.resumeRetryCount="-1"
        queue.saveonshutdown="on"
        Target="10.11.49.116" Port="514" Protocol="tcp")
```

c) Haga todo tipo de propuestas sobre los siguientes aspectos.: ¿Qué problemas de seguridad identifica en los dos apartados anteriores? ¿Cómo podría solucionar los problemas identificados?

¿Qué problemas de seguridad identifica en los dos apartados anteriores?

Apartado A:

En el apartado A algunos de los problemas que podríamos tener serían, los siguientes, las conexiones NTP, utilizan el protocolo UDP que está siendo muy usado para los ataques de denegación de servicio, es decir estos ataques se basan en la falsificación del tráfico IP (spoofing). Otro posible problema es la posibilidad de que un equipo genere tráfico con una IP que no le corresponde y que éste llegue hasta internet. Recordemos también que con UDP podemos perder paquetes muy fácilmente. En resumen, lo que puede llegar a pasar sería que en las versiones antiguas de NTP algunas redes tienen además un servicio de monitorización que permite a los administradores recopilar una lista de los 600 hosts que se han conectado al servidor.

Los ciberatacantes aprovechan esta característica haciendo un ataque reflejo: envían un paquete con una dirección IP falsa mediante la que obtienen la lista. Después, lo amplifican realizando un ataque de denegación de servicio (DDoS) que puede dejar en fuera de juego temporalmente la conexión de todas las direcciones de hosts de la lista.

Apartado B:

En el apartado B algunos de los problemas que tendríamos serían que, si hackean el servidor de logs, pueden ver todos los logs de la red. También tenemos la posibilidad de un ataque DDoS, como se explicó en la sección anterior No hay autenticación, permitiendo que cualquiera envíe logs al servidor, no existe confidencialidad de los datos enviados, no hay integridad de los datos (alguien podría modificarlos en el camino – MITM).

Aunque probablemente su problema más grave es la pobre seguridad que provee.

¿Cómo podría solucionar los problemas identificados?

Apartado A

Para mitigar un poco estos ataques se pueden aplicar medidas que limiten el tráfico NTP en las conexiones, pero se requiere que haya definido una jerarquía de servidores a nivel interno de la organización de forma el número de equipos que realizan consultas NTP al exterior sea reducido (restringir el acceso). Una buena manera para limitar el acceso es utilizando un firewall rechazando así todo tipo de conexión que no venga de una IP autorizada.

También deberíamos tener en cuenta otros aspectos como son, una correcta configuración del servidor para no correr el riesgo de que nos cojan nuestra lista de hosts. Otra solución sería la actualización de las redes y sus protocolos, estos ataques a NTP se dan en versiones antiguas y no tanto en las recientes. Y finalmente además de todo lo anterior, es imprescindible que las empresas de cualquier tamaño cuenten con una solución de ciberseguridad avanzada, activa en todos los endpoints y que sea capaz de prevenir, detectar y neutralizar los ataques en todo momento.

Aparatado B

Ya utilizamos TCP para no perder paquetes. La IPsec (abreviatura de Internet Protocol security) es un conjunto de protocolos cuya función es asegurar las comunicaciones sobre el Protocolo de Internet (IP) autenticando y/o cifrando cada paquete IP en un flujo de datos. IPsec también incluye protocolos para el establecimiento de claves de cifrado.

E de igual forma que en el apartado A podemos solucionarlo con un firewall rechazando así todo tipo de conexión que no venga de una IP autorizada.

d) En la plataforma de virtualización corren, entre otros equipos, más de 200 máquinas virtuales para LSI. Como los recursos son limitados, y el disco duro también, identifique todas aquellas acciones que pueda hacer para reducir el espacio de disco ocupado.

Ejecutamos:

- apt clean: Para borrar la cache del sistema.
- apt autoclean: Para borrar paquetes de versiones antiguas.
- apt autoremove: Para borrar sin padres que nunca se ejecutaría.
- rm -rf /usr/share/man: Para borrar los manuales del SO.

Luego para una liberación de memoria más exhaustiva podemos eliminar versiones de kernel antiguas.

Primero verificamos nuestra versión actual de kernel con:

- uname -r

Luego revisamos cuales kernels hay instalados en nuestro sistema con:

- dpkg –list | grep linux-image

Para finalmente eliminarlos con el comando:

- sudo apt purge <kernel a eliminar>