DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 1 - Virtualización

Enunciados de laboratorios
Tecnologías Avanzadas de la Información
Rev. 6 - 25/10/23

1. Introducción y objetivos

La duración estimada de esta sesión de laboratorio es de **4 horas**. El propósito general de esta sesión de laboratorio es familiarizarse con un entorno de trabajo basado en virtualización para ser utilizado a lo largo del curso. Los principales objetivos se resumen como sigue:

- Administración de máquinas virtuales basadas en Linux para la realización de las prácticas de laboratorio.
- Uso de los comandos básicos para administración de red en Linux.
- Establecer una configuración adecuada de red para cada una de las máquinas.
- Mostrar el proceso de instalación de software adicional en las distribuciones Linux.

2. Virtualización

Actualmente la tecnología de virtualización está presente en todos los proveedores de servicios de Internet. Aunque existen gran variedad de soluciones, todas se basan en la capacidad de los microprocesadores actuales de ofrecer tecnología de virtualización a nivel de hardware. En los laboratorios de esta asignatura se usará la virtualización para simular un entorno de red con al menos 3 equipos. Se dotará a los alumnos de las máquinas usando un entorno virtualizado disponible en servidores de los laboratorios.

Para el uso y administrados de las máquinas virtuales se recomienda la utilización de alguna distribución de Linux, aunque es posible el uso de MsWindows. Si usa MsWindows para realizarlos podrá encontrar dificultades al realizar alguna de las tareas al ser necesario realizar diversos tipos de conexiones de red entre el equipo local y la máquina virtual. Para facilitar la realización con MsWindows desde su propio equipo y

en modo remoto será necesario la instalación de software adicional en su propio equipo. Según avancen los laboratorios se mostrarán los programas necesarios y el procedimiento de instalación y configuración.

Programa	Descripción
virt-viewer	Visor de escritorio remoto para MsWindows con funcionalidad para copiar y pegar texto.

Tabla 1. Programas recomendados para la realización del laboratorio.

2.1. Proxmox

El entorno utilizado para dotar a los estudiantes de los recursos virtuales (máquina, redes, almacenamiento, etc) en la asignatura es *Proxmox* [PROXMOX]. Este gestor es uno de las múltiples soluciones integrales existentes para servidores dedicados a la virtualización. Sus principales características con el soporte de varios de los tipos de virtualización y el hecho de que se distribuye como una distribución Linux independiente. En realidad es un derivado de la distribución Debian y, de igual modo que esta última, está desarrollada bajo una licencia copyleft (GNU AGPL) por lo que se puede considerar software libre/abierto. Para facilitar la gestión y uso remoto, Proxmox dispone de una consola Web desde la que se pueden administrar todos los recursos de virtualización, donde además permite gestionar usuarios, grupos, permisos y roles de forma que se puede delegar recursos (máquinas almacenamientos) en usuarios y grupos creados para tal fin.

En este curso cada estudiante dispondrá de un usuario con acceso al servidor Proxmox de la asignatura. Se ha dotado a cada usuario de tres máquinas virtuales con un sistema operativo preinstalado, Debian 11. Todos las máquinas partirán de una plantilla idéntica con la distribución Debian preinstalada y cada alumno deberá configurar adecuadamente sus máquinas para preparar el entorno de trabajo.

Tarea 1.- Acceda al enlace proporcionado por el profesor donde está del panel de control de los laboratorios de la asignatura. El primer paso será reiniciar su contraseña en este sistema para que se activen todos recursos de la asignatura. Después navegue a la información del Laboratorio 1.

T1.1.- Acceda al servidor de virtualización, el enlace lo encontrará con la información del laboratorio. EL servidor de virtualización dispone de una Web separada donde tiene que volcer a identificarse. Inicie la sesión seleccionando la opción mostrada en la figura 1. Para poder acceder debe reiniciar su clave desde el panel de control de los laboratorios de la asignatura.

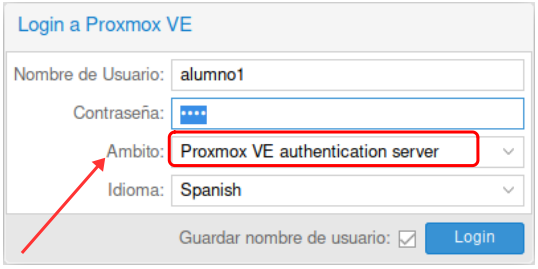


Figura 1. Inicio de sesión en Proxmox.

T1.2.- Una vez entre en Proxmox, navegue por las opciones hasta encontrar el listado de máquinas virtuales asignadas a su usuario. El listado se mostrará de manera similar a la captura de pantalla mostrada en la figura 2.

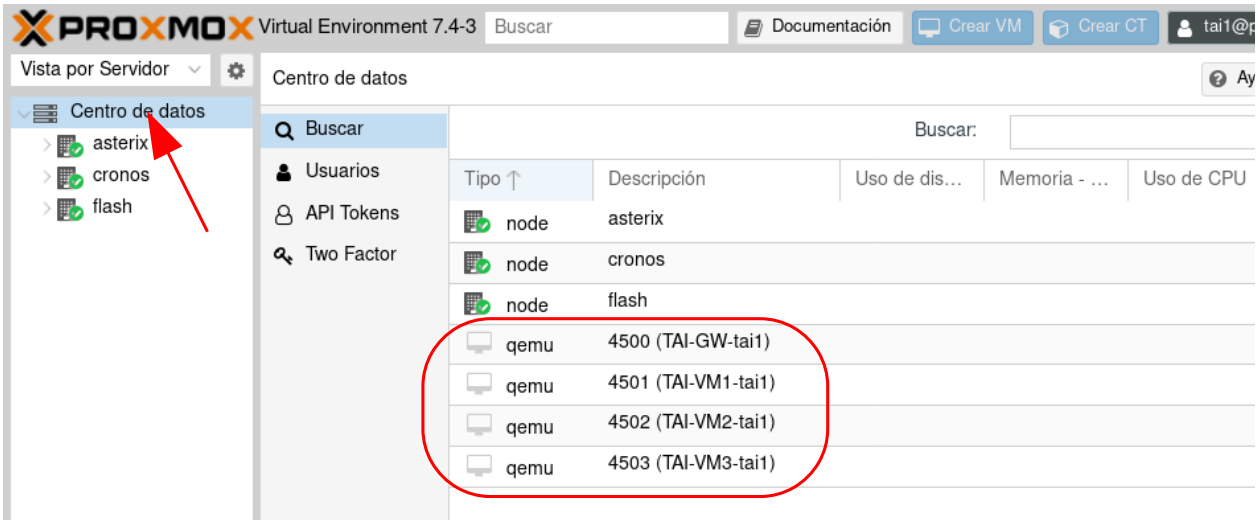


Figura 2. Máquinas virtuales disponibles para cada usuario.

T1.3.- Para cada máquina, abra el panel con información de la misma y busque un panel titulado Hardware con el listado de dispositivos de la máquina. ¿Qué máquina de las 4 disponibles tiene 2 dispositivos de red?

T1.4.- Inicie la máquina que tiene 2 dispositivos de red. Para conseguirlo siga la secuencia mostrada en la figura 3, se recomienda esperar unos segundos antes de acceder a la consola. Si está usando un equipo con MsWindows debe instalar el software de tabla 1 para ver la pantalla remota correctamente. Hay un enlace directo a este programa en el panel de control de la asignatura.

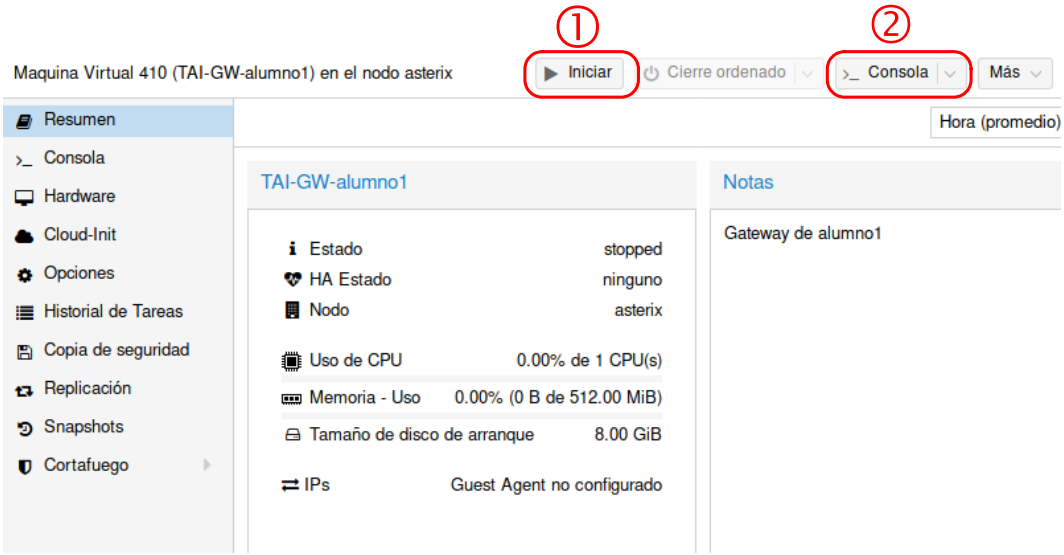


Figura 3. Inicio de una máquina.

T1.5.- Inicie una sesión en la consola de la máquina para comprobar el correcto funcionamiento del

sistema operativo. Use el usuario y la contraseña en la página Web con la información sobre el Laboratorio.

T1.6.- Pruebe apagar la máquina usando el botón del panel de control del menú Cierre ordenado.

3. Uso básico de Debian Linux

Una vez mostrado el procedimiento de conexión al servidor de virtualización se mostrará una introducción al uso de las máquinas asignadas. La distribución Linux usada en este curso es Debian GNU/Linux y la imagen suministrada es una instalación mínima de esta distribución. El punto inicio para cualquier distribución Linux es la línea de comandos. Esta interfaz de comandos es muy avanzada y facilita en gran medida las tareas de administración. Durante este curso se irán presentando los diferentes comandos necesarios para realizar la tareas solicitadas, pero antes de continuar presentaremos algunos comandos básicos. Cabe destacar el primero de todos, **man,** el cual sirve para mostrar ayuda sobre cualquier comando disponible.

Comando	Descripción
man	Muestra ayuda
ls	Listado de fichero
mkdir / rmdir	Manipulación de directorios
cp / mv / rm	Manipulación de dicheros
sudo	Ejecutar comando como administrador (root)
nano	Editor de textos
aptitude	Gestor de paquetes para instalación de software adicional.
exit	Cerrar el shell actual

Tabla 2. Comandos básicos con Debian GNU-Linux

Tarea 2.- Inicie de nuevo la máquina GW y acceda a la consola, verá el texto login esperando un nombre de usuario. Acceda con el usuario y la contraseña proporcionadas para probar los siguientes comandos:

T2.1.- Comando para solicitar ayuda sobre el comando *su*: `man sudo`. Para salir de la ayuda use la tecla q.

T2.2.- Comando para convertirse en administrador del equipo: `sudo su` (se le solicitará de nuevo la contraseña).

T2.3.- Abandone la sesión utilizando el comando `exit`.

T2.4.- El comando `systemctl reboot` reinicia la máquina, intente ejecutarlo como un usuario no administrador y observe el error. Pruebe ejecutarlo como administrador mediante el comando `sudo systemctl reboot`, o simplemente escribir `reboot` como administrador.

T2.5.- Ejecute el comando id para ver el usuario que es en cada momento.

T2.6.- De nuevo como administrador (usuario root) ejecute simplemente el comando `reboot`.

T2.7.- Una vez reiniciada, como administrador (usuario root) ejecute el comando `poweroff`.

4. Configuración de la red de máquinas virtuales

Con las tres máquinas virtuales asignadas, el objetivo es conseguir una configuración de red como la mostrada en la figura 4. Las tres máquinas estarán conectadas a una red virtual privada de cada alumno (192.168.7.0/24) y una de ellas hará de puerta de enlace (gateway) hacia la red externa. La máquina gateway (GW) dispone de interfaces de red, una conectada a la red virtual interna y otra conectada a la red de laboratorio donde se puede utilizar la red 192.168.20.0/24. Toda la configuración hardware está realizada internamente en Proxmox, pero la asignación de direcciones IPs a las máquinas debe realizarla de forma correcta cada alumno.

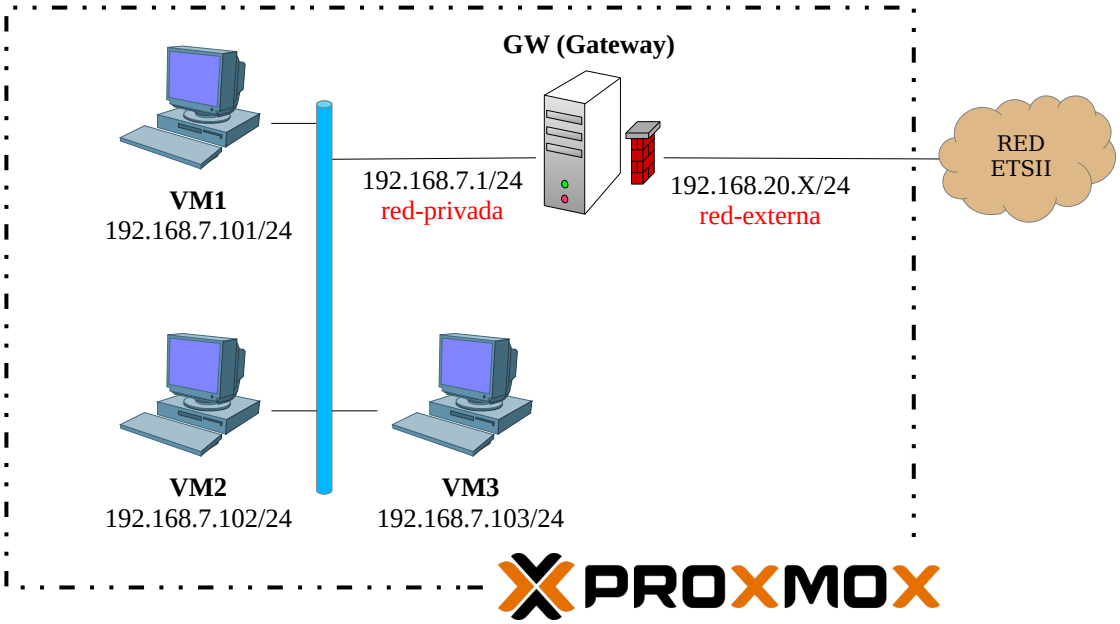


Figura 4. Esquema de configuración de la red virtual.

Las máquinas suministradas tienen establecido el nombre a GW, VM1, VM2 y VM3 haciendo referencia a las cuatro máquinas de la figura 4. Así las máquinas VM1 y VM2 tienen el mismo hardware, con una única interfaz de red. VM3 también tiene una interfaz de red pero dispone de más de 1GB de RAM. En cambio la máquina gateway (GW) dispone de 2 dispositivos de red. Todos los dispositivos de red de todas las máquinas están sin configurar, y será objetivo en esta asignatura configurar correctamente toda la red para que la máquina puerta de enlace (GW) opere correctamente.

La configuración de la red se realizará en dos bloques, en primer lugar la red privada que es la misma para todos los alumnos y después se conectará la máquina GW a la red externa y se comprobará si se dispone de conexión a la red exterior e Internet.

4.1. Configuración de la red privada

Para que el entorno de trabajo opere correctamente se deben configurar las interfaces de red de todas las máquinas virtuales siguiendo el esquema de la figura 4. En las distribuciones Linux la gestión de la red se puede realizar con diferentes administradores de red: *NetworkManagar*, *Systemd*, *Dhcpcd*, *Wicd*, etc. Para simplificar la configuración se utilizará el método clásico usado en *Debian*, consistente en la modificación de ficheros de texto (puede obtener más información en [NetworkConfiguration]). Los ficheros de texto relevantes para la configuración manual de la red son:

- /etc/network/interfaces: Fichero con configuración explícita para cada interfaz de red disponible.
- /etc/hostname: Nombre de la máquina.
- /etc/hosts: Fichero con las correspondencias de nombres de máquinas y direcciones IP, el cual es prioritario frente a la resolución de nombres mediante DNS.

Además de los ficheros enumerados, se dispone de una serie de comandos para consultar o cambiar la configuración de red en todo momento.

Comandos	Descripción
ifup ifdown ifquery	Configuración explícita de la red basado en el fichero */etc/network/interfaces*.
ping	Envío de paquetes ICMP ECHO_REQUEST.
traceroute	Seguimiento de ruta y saltos.
ip	Comando avanzado para manipulación de red.

Tabla 3. Comandos básicos para la manipulación de red en GNU-Linux.

Tarea 3.- La primera tarea es cambiar el nombre de *Host* en cada una de las máquinas virtuales para evitar conflictos de nombres puesto que, al haber sido clonadas desde una plantilla, todas tienen el mismo nombre. Hay que editar dos ficheros: /etc/hostname y /etc/hosts y debe utilizar los comandos indicados en cada una de las máquinas cambiando adecuadamente el nombre de Host en cada una de ellas:

T3.1.- Utilice el comando `sudo su` para convertirse en administrador (root). Para asegurarse que es administrador (root) ejecute también el comando `id`.

T3.2.- Ahora edite el fichero /etc/hostname usando el comando `nano /etc/hostname`. Para la máquina GW utilice su *uvus* como parte del nombre de la máquina, de forma que quede *gwuvus (*no debe usar espacios ni otros caracteres que no sean letras y guiones*)*. En la figura 5 se muestra una captura del editor de textos *nano* y en la parte inferior se muestras las teclas de acción precedidas del símbolo ^. Este símbolo hace referencia a la tecla CTRL, por ejemplo, para guardar los cambios en el fichero debe pulsar CTRL+O (ambas teclas simultáneamente) y para salir del editor CTRL+X.

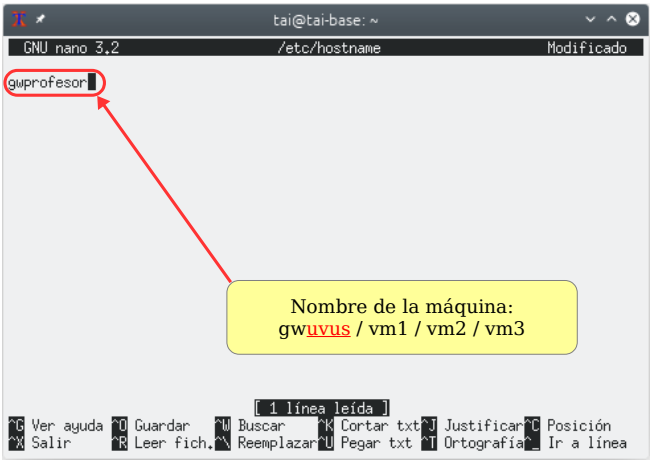


Figura 5. Edición del fichero hostname.

T3.3.- Debe repetir la edición pero con el fichero `/etc/hosts` y establecer el nombre cada máquina en la línea mostrada en la figura 6, no elimine la primera línea donde es establece el nombre para *localhost*.

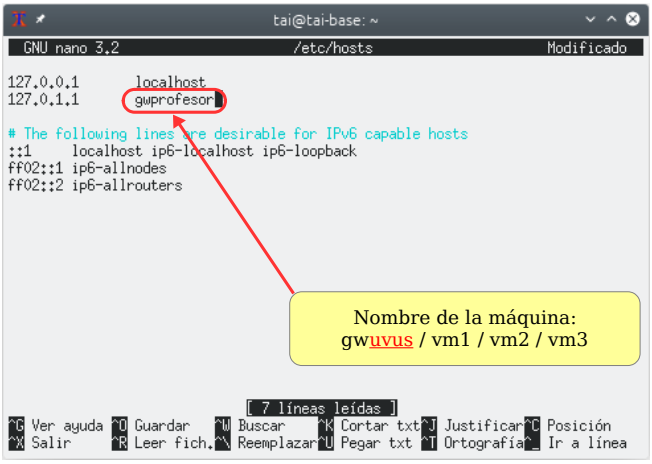


Figura 6. Edición del fichero hosts.

T3.4.- Para hacer los cambios efectivos reinicie la máquina usando el comando `reboot` como administrador. Una vez reiniciada compruebe los cambios usando los comandos `hostname`

T3.5.- Debe repetir el proceso para las otras tres máquinas, estableciendo los nombres *vm1, vm2 y vm3.*

El siguiente paso es configurar la red en las diferentes máquinas, pero hay que distinguir entre las máquinas *VM1*, *VM2* y *VM3* de la máquina *Gateway (GW)*, la cual, posee dos interfaces de red. Como las primeras sólo tienen un único adaptador de red la mejor opción es empezar con ellas.

Tarea 4.- Para configurar la red de las máquinas VM1, VM2 y VM3 debe iniciarlas y acceder a la consola Web. Cuando haya iniciado las consolas de las máquinas, debe realizar estos pasos en cada una de las máquinas:

T4.1.- Comenzando en VM1, acceda como usuario *tai* para luego convertirse en administrador (root) mediante el comando `sudo su`. Introduzca la clave adecuada.

T4.2.- Ejecute el comando `ip a` para ver los nombres de las interfaces y la configuración de la red, obtendrá una salida como la mostrada, siendo las interfaces de red las marcadas en rojo:

```
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
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 08:00:27:43:0a:b2 brd ff:ff:ff:ff:ff:ff
```

Código 1. Salida del comando ip.

T4.3.- Supongamos que nos aparecen las interfaces *eth0* y *lo*. La interfaz *lo* corresponde al bucle local (localhost/127.0.0.1) y *ethX* a las tarjetas de red (donde X es un número). *Entre en el directorio* /etc/network/ *mediante el comando* `cd /etc/network/`. Liste el contenido del mismo con `ls -l` y con un editor de texto, por ejemplo *nano*, edite el archivo de este directorio llamado interfaces, utilice el comando `nano /etc/network/interfaces` y realice la siguiente configuración:

```
auto lo
    iface lo inet loopback

auto eth0
iface eth0 inet static
    address 192.168.7.101
    netmask 255.255.255.0
    gateway 192.168.7.1
```

Código 2. Contenido de /etc/network/interfaces para configuración con IP estática.

T4.4.- Tras guardar los cambios en el fichero debe comprobar si no ha cometido errores se sintaxis en el fichero. Para ello ejecute el comando `ifquery eth0`, en caso de error le indicará el número de línea.

T4.5.- Si el comando anterior mostró correctamente la configuración de red, puede reiniciar la configuración de red utilizando los comandos: `ifdown -a` seguido de `ifup -a`. Para comprobar la nueva configuración use los comandos `ip a` y `ip route`.

T4.6.- Ejecute el comando `man interfaces` para consultar la documentación sobre este fichero.

T4.7.- Repita los pasos en la máquina VM2 estableciendo la dirección IP 192.168.7.102 y en la máquina VM3 con la IP 192.168.7.103.

T4.8.- Cuando tenga configuradas las tres máquinas ejecute el comando *ping* entre ellas para verificar el correcto funcionamiento de la red interna. El comando sería desde VM1 `ping 192.168.7.102` y desde VM2 `ping 192.168.7.101`. También debe comprobar VM3.

T4.9.- El comando *ping* nunca termina, envía paquetes ICMP de manera continuada e infinita ¿Cual es la combinación de teclas usado para parar cualquier comando que se está ejecutando en una consola

Linux?. Debe los comandos forma correcta.

4.2. Configuración de la máquina Gateway

La máquina *Gateway* es diferente de las anteriores, dispone de dos adaptadores de red. Estos adaptadores de red son virtuales, no hay un cable físico detrás. En este contexto, al no disponer físicamente de conexiones para poder identificarlas, la única opción es consultar y apuntar las direcciones MAC de cada adaptador para, posteriormente saber cual es el que está conectado a la red privada y cual a la externa (figura 4). Para identificar los adaptadores hay que obtener la dirección MAC de cada adaptador desde la línea de comandos de la máquina virtual. Esto se hace analizando la salida del comando `ip a` y después, desde el panel de control Web de Proxmox, identificar las redes a las que está conectado cada adaptador. Todo este proceso se realiza a continuación.

Tarea 5.- Acceda a una consola en la máquina gateway (GW), conviértase en administrador y ejecute el comando `ip a` para identificar los adaptadores de red, observará una salida parecida a la siguiente:

```
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
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 6a:4f:90:de:4e:2c brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether ca:ac:2d:eb:d1:c2 brd ff:ff:ff:ff:ff:ff
```

Código 3. Salida del comando ip addr en la máquina gateway.

T5.1.- Debe comparar las direcciones MAC de cada adaptador eth0 y eth1 (resaltadas en azul) con las mostradas en el panel de control de Proxmox. Fíjese en el código 3 la secuencia resaltada en azul para cada interfaz, y compárela con la dirección hardware (MAC) mostrada en el panel Hardware de Proxmox (ver figura 7). La **red externa** es la resaltada en la figura 7 porque está asociada al puente **bridge=vmbr20**, por tanto la **red privada** será el otro adaptador.

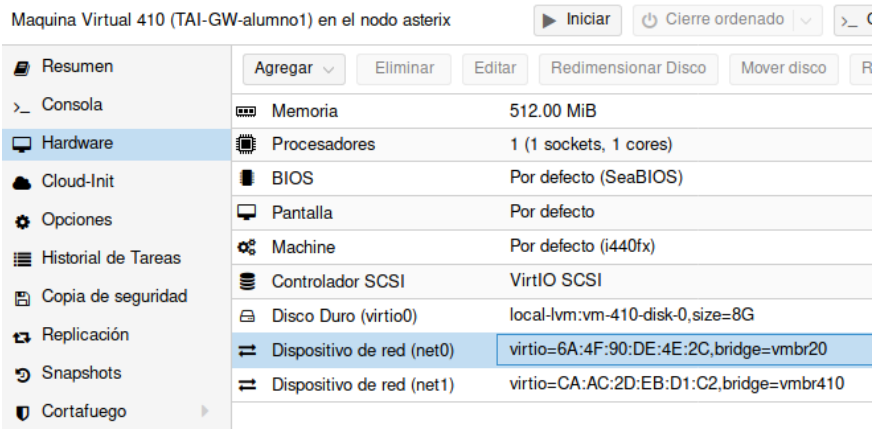


Figura 7. Dirección MAC de los adaptadores de red en Proxmox para la máquina gateway.

T5.2.- Una vez identificada la **red privada**, establezca la configuración **únicamente para esta interfaz**

repitiendo los pasos desde T4.1.- a T4.4.- usando la dirección IP 192.168.7.1. ¿Qué interfaz corresponde a la red interna *eth0* o *eth1*?

T5.3.- Pruebe con las cuatro máquinas virtuales el comando *ping* entre todas ellas. Las máquinas VM1, VM2 y VM3 deben responder correctamente el comando `ping 192.168.7.1`.

En la configuración de red de sus máquinas habrá observado que existen dos redes virtuales construidas mediante *puentes de red* (redes virtuales en Proxmox). Los puentes virtuales de red simplemente conectan los diferentes adaptadores de red que estén incluidos en dicho puente. En el caso de esta asignatura, el puente *vmbr20* conecta todas máquinas GW de todos alumnos de curso, así, todas las máquinas GW están en la misma red virtual. Sin embargo, el otro puente con la numeración *vmbrXXX* sólo une los 4 adaptadores de cada alumno, de este modo, cada alumno tiene su propia red virtual privada.

Tarea 6.- La interfaz externa debe configurarse con una dirección IP proporcionada con el material de la asignatura para evitar conflictos con los compañeros de clase que estarán en la misma red. Esta red será 192.168.20.0/24.

T6.1.- En el panel de control Web de la asignatura está la página de recursos asignados por alumno. Cada alumno tiene asociada una dirección IP, debe acceder a ella y consultarla. Edite de nuevo el fichero `/etc/network/interfaces` añadiendo la configuración de la interfaz externa identificada en la tarea anterior. La configuración es como la indicada donde debe sustituir `ethX` y `192.168.20.YYY` por los valores correctos.

```
# Sustituya ethX y 192.168.20.YYY por el valor correcto
auto ethX
iface ethX inet static
    address 192.168.20.YYY
    netmask 255.255.255.0
    gateway 192.168.20.1
    dns-nameservers 150.214.130.15 150.214.5.83
```

Código 4. Configuración de la interfaz externa de red en el gateway.

T6.2.- Cuidado en este momento, puede que tenga en el archivo de configuración dos gateways, para esta máquina, su gateway es únicamente 192.168.20.1, por ello elimine de la configuración del otro interfaz la sentencia `gateway 192.168.7.1`.

T6.3.- Compruebe la configuración con el comando `ifquery` y si es correcto, reinicie la configuración de red los siguientes comandos: `ifdown -a` (desactiva todas las interfaces) `ifup -a` (activa todas las interfaces). Compruebe la nueva configuración mediante los comandos `ip a` e `ip route`.

T6.4.- Para comprobar si tiene conexión a Internet utilice el comando *ping*, primero compruebe que que alcanza el gateway externo 192.168.20.1 y en caso afirmativo, compruebe si llega al servidor DNS 150.214.130.15.

T6.5.- Si en el paso anterior obtuvo una respuesta correcta con el comando ping desde el servidor DNS, puede comprobar la resolución de nombres. La resolución de nombres se puede revisar usando el comando *host* o el mismo comando *ping*. Pruebe el siguiente comando `host www.dte.us.es` y después `ping www.dte.us.es`. ¿Coincide la dirección IP resuelta por el comando *host* con la dirección IP usada

en el comando *ping*?

Llegado a este punto tiene la red interna y externa configuradas correctamente en la máquina GW, pero podrá comprobar que aunque los equipos de la red interna (VM1, VM2 y VM3) alcanzan la máquina GW, estas máquinas no consiguen salir al exterior a través de su máquina *Gateway*, por no estar activado y configurado el reenvío (*forward*) Para conseguir que las máquinas internas salgan a Internet será necesario configurar adecuadamente NAT y el firewall, lo cual, se realizará en la siguiente sesión de laboratorio.

Tarea 7.- Visualice la ruta de los paquetes en su máquina *Gateway* hasta los DNS públicos de Cloudflare mediante el comando `traceroute 1.1.1.1`.

T7.1.- Repita el proceso desde las máquinas internas de red para comprobar que su gateway no retransmite los paquetes al exterior.

5. Instalación de aplicaciones

En Linux existen varias formas de instalar un programa en función del formato disponible del programa, cada una de ellas presenta las siguientes peculiaridades:

- Binarios: Generan problemas de compatibilidad de dependencias y requieren actualizaciones manuales. No se utilizan habitualmente.
- Código fuente: Se necesitan entornos de desarrollo y bibliotecas y consiste en compilar el código fuente original. También se es necesario resolver manualmente las dependencias. En caso de actualizaciones hay que repetir el proceso.
- Paquetes de la distribución: Es la opción mas recomendable, es fácil, rápido, automático, centralizado, etc. Además los procesos de actualización están automatizados.

Otro concepto importante para la instalación de programas en un sistema Linux son las *dependencias*. Dependencia significa que un software necesita de otro para que funcione adecuadamente. En Linux es común que se necesiten herramientas o librerías para realizar un trabajo. Este problema se puede resolver, en parte, con programas que se encargan del software instalado y que tratan de resolver las dependencias con información proveída por personas encargadas de los paquetes. La resolución automática de dependencias es una de las tareas más importantes de una distribución.

Las distribuciones modernas de Linux utilizan los llamados *repositorios de paquetes* para facilitar al usuario la adquisición y descarga del software adicional. Un repositorio es un lugar físico (servidor) donde se encuentran paquetes de software de la distribución. En un repositorio puede haber varias versiones de una distribución. Por ejemplo, en el repositorio de Debian podemos encontrar: Versiones soportadas anteriormente, Versión actual y Versión de desarrollo.

También para cada versión de la distribución suelen existir varios componentes por motivos diversos. Por ejemplo, en Debian encontramos estas secciones:

- main: sección principal, libre y con soporte oficial.
- restricted: software soportado cuya licencia no es completamente libre.

- universe: software libre adicional, no soportado oficialmente.
- multiverse: software no soportado oficialmente con posibles problemas de distribución y no libre.

Pueden combinarse cualquier número de repositorios, siempre que no existan conflictos entre los paquetes que los componen. El sistema de gestión de paquetes elige la versión mas moderna en caso de paquetes repetidos.

Este sistema de paquetes es el más utilizado actualmente y el más fácil de manejar, ya que existen multitud de herramientas, tanto de consola como gráficas, con multitud de opciones. Desde el punto de vista de la línea de comandos, los comandos básicos para gestión del software en Debian son las mostradas en la tabla 4, de los cuales, se recomienda centrarse en *aptitude*.

Comando	Descripción
apt	Simplificación de alto nivel para el gestor de paquetes.
apt-get	Herramienta completa de gestión de paquetes APT.
apt-cache	Búsqueda de paquetes.
aptitude	Interfaz amigable basada en *ncurses*.
apt-file	Búsqueda de un fichero en el repositorio de paquetes.
dpkg	Gestor de paquetes de bajo nivel de Debian.

Tabla 4. Comandos básicos de gestión de paquetes.

5.1. Instalación de software adicional

En esta sección se propone instalar software en la máquina GW, puesto que es la única con conexión a Internet para descargar paquetes con APT. Pero antes de nada, es importante a lo largo de todo el curso mantener actualizadas las máquinas con las últimas versiones de todos los programas.

Tarea 8.- Antes de instalar software adicional es siempre necesario actualizar la base de datos de los paquetes disponibles en la red y descargar las últimas actualizaciones. Los comandos mostrados a continuación debe usarlos en la máquina GW y esperar a que la actualización termine:

```
apt update
apt upgrade
```

Código 5. Actualización de software con APT.

T8.1.- Una vez terminada la actualización reinicie la máquina GW usando el comando `reboot` como administrador.

T8.2.- De nuevo como administrador, instale el paquete *mtr* mediante `apt install mtr` y pruebe el comando hacia una dirección externa de la Universidad, por ejemplo `mtr 8.8.8.8` ¿Para qué sirve este comando?

T8.3.- Ejecute el comando `apt-cache show mtr` para obtener toda la información sobre este programa.

Se ha presentado la instalación de software mediante APT usando la línea de comandos. Si se fija en todos los ejemplo usados hasta el momento, siempre ha sido necesario conocer exactamente el nombre del paquete (programa) se que desea instalar, de lo contrario, el comando fallará. El uso de APT en línea de comandos permite buscar usando el comando *apt-cache*, pero su salida es demasiado extensa. Se propone a continuación instalar un gestor de paquetes que permiten buscar y ver los listados de paquetes instalados y no instalados.

Tarea 9.- Se instalará el gestor de paquetes *aptitude* que funciona en modo texto el cual tiene múltiples opciones y menús. De nuevo en un terminal conviértase en administrador para instalar el paquete *aptitude* con APT.

T9.1.- Cuando finalice la instalación ejecute el comando `aptitude` y pulse la tecla F10 para acceder a los menús del programa donde debe usar los cursores para moverse.

T9.2.- Acceda al menú buscar y busque el paquete *tcpdump*. Una vez encontrado pulse la tecla Entrar y verá los detalles de este programa.

T9.3.- Este programa será útil para su uso a lo largo de los laboratorios. Marque este paquete usando acceduendo al menú Paquete → Instalar, esto hace que se marque para la instalación. Para ejecutar la instalación de todos lo paquetes marcados usel el menú Acciones → Instalar/Eliminar paquetes.

5.2. Conexión remota a las máquinas

El objetivo de esta sección es conectar a las máquinas virtuales de una manera adecuada cuando se realizan tareas de administración. Siempre que se sea posible se recomienda evitar el escritorio remoto, puesto que en servidores reales nunca se dispone de un escritorio remoto y todo se administra desde la línea de comandos mediante conexión SSH [RFC4250].

Las tareas propuestas a continuación deben ser realizadas desde los equipos de laboratorio del aula ya que debe estar accesible la red 192.168.20.0/24 desde el equipo local en el cual trabaja. Alternativamente, para poder trabajar de forma remota, tanto con Linux como con MsWindows se debe proceder a instalar en un equipo personal una VPN proporcionada para la asignatura. Con ella, podrá para conectar su equipo personal desde cualquier ubicación. Si no está trabajando desde los equipos del aula, proceda a instalar y activar la VPN siguiendo las instrucciones proporcionadas en el panel de control de la asignatura.

Tarea 10.- En su equipo local, sea Linux o MsWindows, abra un terminal (comando `cmd` en MsWindows). Si está usando MsWindows asegúrese de tener activa la VPN de la asignatura.

T10.1.- Para comprobar que está accesible la red de la asignatura, desde el terminal testee la IP 192.168.20.1 mediante `ping 192.168.20.1`. Si no obtiene respuesta no podrá seguir.

T10.2.- Encienda su máquina virtual y compruebe si su máquina es alcanzable mediante `ping 192.168.20.X` (sustituyendo X la IP asignada durante el curso).

T10.3.- El siguiente paso es conectar mediante SSH, para obtener una consola de comandos en el

ordenador remoto. En la consola local ejecute `ssh tai@192.168.20.X`, acepte las credenciales remotas e introduzca la contraseña cuando se lo solicite.

T10.4.- Una vez dentro de su máquina, conviértase en administrador y ejecute el comando `reboot`. ¿Qué ha mostrado la consola tras reiniciarse la máquina remota? ¿Puede volver a iniciar la conexión mediante SSH?. Cuando lo consiga pruebe el comando `uptime` y deduzca que información le está mostrando.

T10.5.- ¿Cuando entró en su máquina mediante `ssh tai@192.168.20.X` que contraseña usó? Si no se cambia la contraseña del usuario *tai*, puede acceder a cualquier máquina de otro compañero sólo cambiando la IP destino. Como **requisito** de este laboratorio debe cambiar la contraseña del usuario *tai* y para proteger su máquina se accesos no controlados.

5.3. Uso de la máquina compartida

A lo largo del curso se realizarán diversas pruebas en la red 192.168.20.0/24 desde equipos externos al de cada alumno. Para no tener que solicitar a algún compañero que realice la prueba se ha preparado una máquina virtual compartida para todos los estudiantes del curso. En esta máquina todos los alumnos tienen disponible una cuenta de usuario con la misma contraseña que la del panel de control. A esta máquina se puede acceder mediante SSH y también está disponible un escritorio remoto, pero no desde la interfaz de Proxmox, sino mediante el protocolo RDP de Microsoft [RDP] por tanto es compatible con el programa de escritorio remoto de MsWindows.

Existen múltiples modos de conexión remota en los sistemas informáticos y Linux ofrece multitud de alternativas. En este sentido se pretende mostrar de otro modo de conexión a escritorio remoto a través de la red, pero no por el navegador Web, sino un escritorio virtual en la máquina remota. Para esto existen multitud de protocolos de escritorio remoto disponibles como con VNC, SPICE, RDP, etc. Sin embargo, como ya se ha dicho, para compatibilizar el escritorio remoto con el sistema MsWindows se ha instalado en la máquina compartida un servidor compatible con el protocolo RDP de Microsoft, así se podrá conectar tanto desde MsWindows como desde Linux sin instalar software adicional.

De nuevo, las tareas propuestas a continuación deben también ser realizadas desde los equipos de laboratorio del aula o en remoto usando la VPN.

Asegúrese de tener accesible la red 192.168.20.0/24 desde el equipo local en el cual trabaja. Para poder trabajar de forma remota, tanto con Linux como con MsWindows se debe proceder a instalar en el equipo local la VPN proporcionada para la asignatura, con ella, podrá para conectar su equipo personal desde cualquier ubicación. Si no está trabajando desde los equipos del aula, proceda a instalar y activar la VPN siguiendo las instrucciones proporcionadas en el panel de control de la asignatura.

Tarea 11.- Asegúrese de tener accesible la red 192.168.20.0/24 desde el equipo local en el cual trabaja. Revise los pasos de la Tarea 10.-

T11.1.- Acceda al panel de control de la signatura para obtener la dirección IP de la máquina compartida. Tanto en Linux como en MsWindows abra una consola e intente acceder por SSH usando su nombre de usuario, el comando sería `ssh uvus@192.168.20.3` pero usando su *uvus* y la IP correcta. Si ha conseguido entrar ejecute el comando `who` para ver quien está en la máquina y la IP desde la que está

conectado. A continuación se muestra un ejemplo con el *uvus alumno1*, en azul están los comandos ejecutados y en rojo el símbolo del sistema (*prompt*) el cual le muestra siempre el usuario y la máquina donde está conectado.

```
ssh alumno1@192.168.20.3

alumno1@192.168.20.3's password:

Linux tai-compartida 5.10.0-8-amd64 #1 SMP Debian 5.10.46-4 (2021-08-03) x86_64

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

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Sep 10 11:48:09 2021 from 192.168.20.1

alumno1@tai-compartida:~$ who

alumno1  pts/0        2021-09-11 13:06 (192.168.22.206)
tai      pts/1        2021-09-11 13:07 (192.168.22.209)

alumno1@tai-compartida:~$
```

T11.2.- Ahora se procederá a conectar al escritorio remoto de la máquina compartida, lo cual es posible tanto desde Linux como desde Windows. Las siguientes tareas están marcadas como Linux o MsWindows, haga las que corresponda al equipo sobre el que esté trabajando.

T11.3.- Linux: Esta prueba se debe realizar desde el equipo Linux del laboratorio y se puede conectar al escritorio con dos programas diferentes. En primer lugar en el equipo local abra un terminal y use el comando `rdesktop 192.168.20.X`, usando la IP de la máquina compartida. Aparecerá una ventana de bienvenida para introducir su *uvus* y la clave correspondiente, debe iniciar la sesión y comprobar que aparece un escritorio.

T11.4.- MsWindows: En el cuadro de búsqueda en la barra de tareas, escriba Conexión a Escritorio remoto y ejecute el programa encontrado. Este programa debe rellenar el cuadro de texto Equipo con la dirección IP de la máquina compartida y pulse el botón Conectar. Aparecerá una ventana de bienvenida para introducir su *uvus* y la clave correspondiente, debe iniciar la sesión y comprobar que aparece un escritorio.

6. Copias de seguridad

Al finalizar cada laboratorio es recomendable realizar una copia de seguridad de la máquinas, pero debe ir borrando alguna de ellas puesto que están limitadas a un máximo de 3 o 4 por máquina. En caso de tener que restaurarlas si tiene dudas, es recomendable contactar con el profesor, aunque a continuación se explica detalladamente el procedimiento para obtener ficheros sin necesidad de realizar una restauración completa.

Tarea 12.- Acceda al panel de control de la máquina Gateway (GW) y apague la máquina.

T12.1.- Acceda al panel Copia de seguridad de esta máquina y pulse sobre el botón Copia de seguridad ahora. Use las opciones de la figura 8 para realizar la copia, en Almacenamiento escoja el que esté

disponible aunque no coincida con el mostrado en la figura, y también, se recomienda usar el Modo Parar. Use el botón Copia de seguridad y espere a que termine.

Figura 8. Realización de copias de seguridad.

T12.2.- Tras finalizar la copia revise el listado de copias de seguridad y aparecerá la nueva copia con la fecha actual.

T12.3.- Si lo cree oportuno, realice copias de seguridad también de las máquinas VM1, VM2, y VM3

La restauración de las copias de seguridad puede ser parcial o total. A continuación sólo se presenta la restauración parcial, ya que la total es improbable que sea necesaria. De todas formas, a lo largo de curso está disponible la restauración total por si fuera necesaria.

Tarea 13.- Acceda al listado de copias de seguridad en el panel de Proxmox y seleccione una de las copias. Use el botón File Restore indicado en la figura 9.

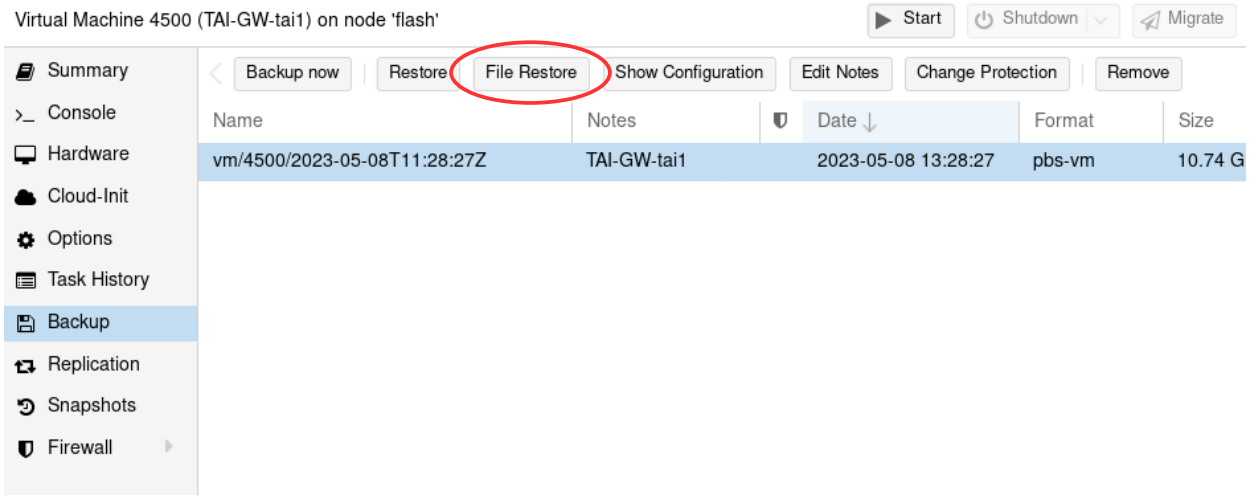


Figura 9. Restauración de ficheros.

T13.1.- Navegue por el los ficheros de la copia de seguridad hasta encontrar el fichero /etc/network/interfaces y descárguelo. Compruebe que el contenido de este fichero corresponde con lo

realizado durante este laboratorio.

Departamento de Tecnología Electrónica
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 2 - Netfilter

Enunciados de laboratorios
Tecnologías Avanzadas de la Información
Rev. 8 - 10/11/23

1. Introducción y objetivos

La duración estimada de esta sesión de laboratorio es de **6 horas**. El propósito general de esta sesión de laboratorio es la configuración de un firewall para el sistema operativo Linux siendo los principales objetivos los siguientes:

- Describir el funcionamiento del filtrado de paquetes.
- Configurar un firewall.
- Configurar filtrado de paquetes en una máquina Linux mediante Netfilter / Nftables.
- Elaborar reglas con nftables.

2. Descripción del firewall

Un cortafuegos o firewall es un sistema o grupo de sistemas utilizados para separar una máquina o una subred (zona protegida) del resto de la red (zona de riesgo), estableciendo una política de control de acceso entre ambas zonas. Es decir, el firewall actúa como punto de interconexión segura entre dos o más sistemas informáticos o redes. En principio, un firewall puede ser un enrutador dedicado, un ordenador de propósito general o incluso una red completa. Un firewall debe ser entendido como un sistema formado por varios elementos, siendo los más comunes un filtro de paquetes y los proxies. En este documento nos centraremos en el primer elemento, el filtrado de paquetes.

El filtrado de paquetes es un proceso que consiste en denegar o permitir el flujo de información entre dos redes no interconectadas a nivel físico. Un escenario habitual consiste en una red interna que se desea proteger y una red externa. El filtrado de paquetes se hace de acuerdo a unas reglas predefinidas, que

examinan las cabeceras de los paquetes según van pasando a través de él y deciden la acción a realizar con el paquete completo (aceptarlo, descartarlo, etc). El filtrado también se conoce como *screening*, y a los dispositivos que lo implementan se les denomina *chokes*. El filtrado de de paquetes actúa sobre la capa de red y la de transporte de la pila TCP/IP. Trabajan sobre la información de las cabeceras de los paquetes IP, sin llegar a analizar los datos. Por ejemplo, el filtrado de paquetes no puede evitar que un usuario de la red sobre la que actúa envíe un correo electrónico desde su equipo hacia una determinada cuenta de correo. Lo que si podría es evitar que dicho equipo accediera al servidor de correo y así no se pudiese enviar ningún correo a nadie. En resumen, este filtrado se realiza a través de una lista de reglas que analizan las cabeceras de los paquetes y decidiendo una acción sobre el paquete que pueden ser aceptación, rechazo o denegación, entre otras.

Las implementaciones de los firewalls de filtrado de paquetes principalmente son de dos tipos: *stateless* y *stateful*. Un filtro estático o *stateless* (sin estado) analiza las cabeceras de cada paquete recibido (IP, TCP, UDP, ICMP, etc.) y toma una decisión de filtrado en función de los valores contenidos en los distintos campos de dichas cabeceras. No se establece ninguna relación entre los diferentes paquetes que atraviesan el filtro, aunque correspondan a una misma conexión. Este mecanismo de filtrado consume pocos recursos y es de fácil implementación. Por el contrario, un filtro dinámico o *stateful* (de estados) permite el control de un flujo de datos relacionados (por ejemplo, paquetes dentro de una misma conexión TCP o entre varias conexiones). Para llevarlo a cabo es necesario mantener en memoria los parámetros de cada conexión, tomando decisiones en función de la evolución de las mismas. Por ejemplo, sólo se permite el paso de datos en sentido entrante a través de un puerto TCP que haya sido previamente abierto en sentido saliente, o conexiones entrantes a un puerto dado desde una dirección origen cuando previamente se ha iniciado una conexión saliente hacia esa misma dirección desde otro puerto concreto. Este último modelo de filtrado es más sofisticado y permite un control más exhaustivo del tráfico, resolviendo necesidades a nivel de paquete que antes tenían que resolverse a nivel de aplicación (mediante *proxies*).

2.1. Network Address Translation (NAT)

NAT (Network Address Translation) es un mecanismo por el cual se alteran las cabeceras de los datagramas IP siendo su uso más común el cambio de las direcciones y puertos destino u origen en el enrutador que lo implementa. Este mecanismo es otra de las funciones básicas incluidas en los sistemas de filtrado de paquetes. En condiciones normales un datagrama IP viaja salto a salto a través de los routers manteniendo en su cabecera las direcciones IP origen y destino durante todo el trayecto. Cuando un router aplica NAT, altera el origen (SNAT) o destino (DNAT) del paquete en uno de los saltos, siendo necesario realizar igualmente la operación inversa en los paquetes de respuesta para que el origen pueda recibirlos. Principalmente la traslación NAT resuelve distintas necesidades:

- Cambio de la dirección y/o puerto destino de los datagramas recibidos en una red, redirigiéndolos a una máquina concreta en función del servicio requerido o distribuyéndolos entre distintas máquinas destino para el equilibrio de carga.
- Cambio de la dirección destino de un datagrama, para que el servicio requerido lo ofrezca un tercer equipo (proxy transparente).
- Cambio de la dirección de origen de un datagrama, asignándole otra dirección antes de reenviarlo

hacia el destino. Este tipo de NAT se denomina enmascaramiento. Permite a uno o varios equipos de una red privada quedar visibles en otra red (por ejemplo Internet) a través de una dirección externa. Esta es la técnica utilizada en la actualidad para que múltiples equipos accedan a Internet a través de una única dirección pública.

Según lo visto podemos distinguir entre:

- Source NAT (SNAT): alteración del origen del datagrama, realizado después del encaminamiento del mismo y antes de su reenvío (el enmascaramiento es una forma de SNAT).
- Destination NAT (DNAT): alteración del destino, realizado antes del encaminamiento del datagrama (el port forwarding, el balanceo de carga y los proxy transparentes son ejemplos de DNAT).

3. Filtrado de paquetes en Linux y Nftables

En Linux, el filtrado de paquetes está programado en el núcleo (como módulo o como componente estático). Los núcleos de Linux han ido evolucionando y con él su firewall, cambiando su implementación en las sucesivas versiones. Así, actualmente se utiliza el módulo *NetFilter* [netfilter.org] como firewall de filtrado de paquetes, el cual, junto con la herramienta *nftables* permite establecer las reglas de filtrado. El entorno Netfilter permite el filtrado de paquetes (ya sea con o sin estado), la traslación de direcciones y puertos (NAT /NAPT) y otras manipulaciones sobre los datagramas.

Netfilter ofrece una interfaz completa (o framework), dentro del núcleo de Linux, que permite interceptar y manipular paquetes de red. A Netfilter pueden conectarse otros módulos o componentes siendo el módulo más conocido *iptables* considerado ya obsoleto en favor de *nftables*. Con *nftables* y una la herramienta en línea de comando llamada *nft*, es posible manipular Netfilter desde el espacio de usuario. Otro módulo construido sobre Netfilter es *conntrack* (Connection Tracking System) o sistema de seguimiento de conexiones. Este último módulo de Netfilter puede funcionar como firewall de inspección de estados (*stateful inspection packet firewall*), también asociado al denominado filtrado dinámico. A través de este mecanismo se puede asociar una cantidad de paquetes a una conexión en particular. Este tipo de inspección permite que el comportamiento del firewall cambie con respecto a la información contenida en el paquete, por ejemplo, de un paquete en particular que es parte de una conexión preexistente.

La herramienta más utilizada para configurar Netfilter ha sido *iptables* la cual está siendo sustituida por *nftables* que es un desarrollo más moderno, eficiente y más fácil de usar. En la mayoría de los tutoriales y documentos existentes en la red se presenta el uso de nftables mediante comparaciones con la herramienta iptables. Puede parecer necesario conocer iptables para usar nftables, pero la realidad es diferente, nftables es mucho más claro en su uso y funcionamiento que iptables, por lo que se ha optado en este documento estudiar nftables sin hacer referencia ni comparación alguna al antiguo iptables, buscando así un aprendizaje más directo, es más, en caso de conocer iptables se recomienda no intentar hacer comparaciones.

A lo largo de los siguientes secciones se trabajará con Netfilter usando nftables donde se presentarán las siguientes funcionalidades:

- Filtrado de paquetes.
- Traducción de direcciones y puertos NAT.

- Manipulaciones sobre datagramas IPv4 e IPv6.
- Seguimiento de conexiones (*conntrack*).
- Registros de logs y depuración de reglas.

Además de la funcionalidad básica incluida con Netfilter, una de las principales características es su diseño modular que permite el uso de plugins adicionales para nuevas condiciones y acciones (Ej: *ipp2p* para filtrar conexiones a redes P2P). Esto, hace que no sea necesario modificar el diseño original para agregar una extensión adicional, pero se necesitan conocimientos más profundos sobre Netfilter para el desarrollo de estos componentes. Aunque existe mucha documentación al respecto, este último aspecto no será tratado en este documento.

Para trabajar con Netfilter y nftables es necesario comprender la organización y relación de todos los elementos que forman parte de este entorno de filtrado de paquetes. En este documento se irán presentando cada una de las partes del sistema junto con ejemplos simples. Las siguientes secciones se dedican a presentar cada parte, siendo el punto de partida son las denominadas *tablas*, las cuales contendrán *cadenas* (*chains*) y éstas cadenas a su vez, están compuestas por *reglas* que se evalúan sobre los paquetes analizados en busca de condiciones, según unos parámetros y, en caso de cumplirse alguna condición, se ejecutarán la acción asociada. Por tanto, tendremos los siguientes elementos: tablas, cadenas, reglas, condiciones y acciones.

3.1. Tablas en Netfilter con Nftables

El primer elemento necesario para poder configurar Netfilter son las tablas. Las tablas son usadas para indicar al sistema de filtrado el tipo de procesamiento que se debe aplicar a los paquetes. Una tabla maneja una cierta cantidad de reglas internas que se organizan en cadenas y cada tabla sólo puede ser de un familia según el tipo de direcciones que maneja:

- Familia *ip* para direcciones IPV4.
- Familia *ip6* para direcciones IPV6.
- Familia *inet* agrupa direcciones IPV4 e IPV6, el objetivo de esta tabla es no tener que repetir reglas IPV4 e IPV6 similares.
- Familia *arp* para direcciones ethernet (capa 2).
- Familia *bridge* para trabajar con los paquetes asociados a puentes.
- Familia *netdev* para poder trabajar directamente con paquetes recibidos por el dispositivo (ingress).

De forma predeterminada no existe ninguna tabla creada cuando se inicia el sistema. Para poder manejar Netfilter con nftables existe una herramienta disponible en la linea de comandos `nft` con una sintaxis que abarca todas las posibilidades. Esta herramienta permite alterar de forma dinámica las reglas de filtrado. Entre las múltiples opciones incluidas en la herramienta, la opción `nft -f` permite cargar un conjunto de reglas de un fichero de texto, así, es posible crear *scripts* ejecutables que carguen toda la configuración de Netfilter poniendo en la cabecera del script: `#!/usr/sbin/nft -f`. A lo largo de este documento se presentarán los ejemplos usando esta posibilidad, de esta forma, se simplifica el uso de líneas de comandos

excesivamente largas.

Comenzando con las tablas, en el ejemplo mostrado en el código 1 se presenta el contenido de un fichero de texto en el que se han creado 2 tablas de diferentes familias vacías, siendo la sintaxis para añadir la tablas es la siguiente:

```
table <familia> <nombre_tabla> { <listado_de_cadenas> }
```

Además de las tablas, fíjese como tras la cabecera puesta en la primera línea `#!/usr/sbin/nft -f` el primer comando pasado es `flush ruleset`. Esto es debido a que nft no borra ninguna regla existente en memoria a no ser que se indique explícitamente. Así, sin este primer comando no se borraría el contenido actual de Netfilter y las tablas mostradas se añadirían tras las ya existentes en memoria.

```
#!/usr/sbin/nft -f
flush ruleset
table inet filtrado_ip {
}
table arp filtrado_mac {
}
```

Código 1. Contenido de un fichero con un ejemplo de creación de tablas.

3.2. Ganchos (hooks) en Netfilter

En Netfilter los paquetes entrantes y salientes se procesan internamente mediante una serie de pasos ordenados que marcan el camino de procesamiento. En algunos de estos pasos es posible enlazar una cadena de reglas del usuario y los lugares donde se puede realizar el enlace se llaman ganchos (*hooks*). Los paquetes sólo se procesan por una cadena de reglas, si esta cadena está enlazada a algún paso sel proceso Netfilter mediante un gancho (*hook*).

En un mismo gancho, pueden estar varias cadenas enlazadas, por ello, en el momento del enlace se debe especificar una prioridad puesto que las cadenas se procesan en el orden especificado por la prioridad. En la figura 1 se muestran en azul los ganchos disponibles y los posibles caminos pueden seguir los paquetes.

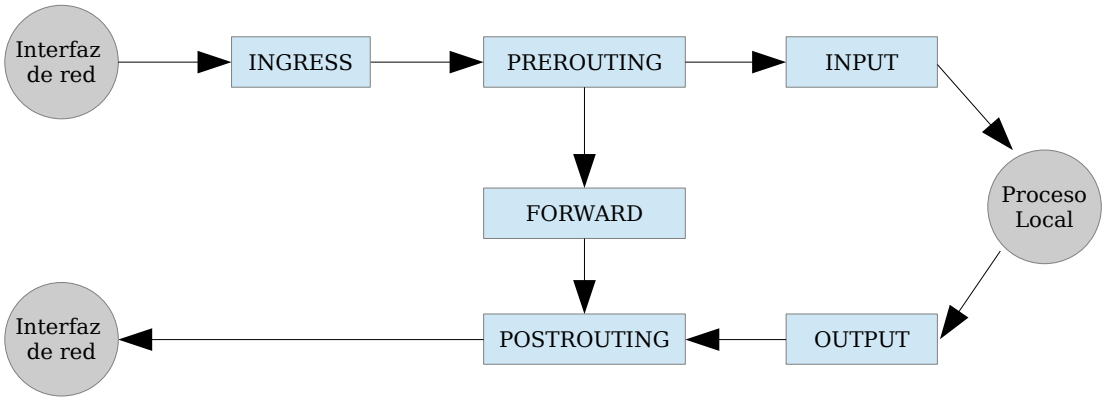


Figura 1. Esquema del procesamiento de NetFilter.

Los ganchos corresponden al paso de procesamiento donde se puede analizar el paquete y actuar sobre él, y su descripción es la siguiente:

- INGRESS: Es el primer gancho disponible en Netfilter. Es un gancho especial de bajo nivel para procesar cualquier paquete entrante y trabaja a nivel de red de forma eficiente. Sólo es usado para aplicaciones muy específicas en entornos muy sobrecargados.
- PREROUTING: Cuando se trabaja con direcciones IPv4/IPv6 es el primer gancho en el sistema Netfilter y determina la primera acción a realizar antes de que el paquete se encamine hacia una ruta determinada, por lo que es el lugar donde, por ejemplo, se puede alterar el destino del paquete.
- INPUT: A este gancho sólo llegan los paquetes entrantes destinados a la propia máquina, es decir, el destino del paquete es alguna de las direcciones IP del propio equipo.
- OUTPUT: Sólo los paquetes originados desde algún proceso (programa) que se ejecuta en el propio equipo son procesados por este gancho. Es decir, se aplica a los paquetes originados en la propia máquina.
- POSTROUTING: Determina la acción a realizar, justo antes de enviar el paquete a la interfaz destino. Por este gancho pasan los tanto los paquetes originados desde el propio equipo, como los que van de camino, en caso de tener la máquina activo el sistema de reenvío de paquetes (*forwarding*).
- FORWARD: En este gancho sólo se procesarán los paquetes que son rutados a otras direcciones que sean la máquina local. Determina la acción a tomar cuando un paquete se envía desde una interfaz a otra. Se trata de una cadena transversal que se encuentra de forma intermedia entre las dos siguientes para distinguir los paquetes no dirigidos al equipo local.

En la figura 1 se puede observar como los paquetes pueden puede recorrer varios caminos, pero los paquetes sólo pueden entrar en Netfilter por dos sitios: por una interfaz de red o un paquete originado por algún proceso (programa) que se esté ejecutando en el sistema. En el primer caso, para los paquetes recibidos por cualquier interfaz, se lleva a cabo en primer lugar una suma de comprobación (*checksum*). Después, si es correcta, los paquetes transitan hacia la evaluación de las cadenas que estén enlazadas al gancho INGRESS y después a las cadenas enlazadas a PREROUTING. En PREROUTING las reglas se encargarán de determinar el tratamiento que deberá de darse al paquete en función de la dirección destino de la siguiente forma:

- Si el paquete va dirigido a la propia máquina, éste es enviado a la cadena INPUT donde pueden existir

más reglas y que en caso de superarse será procesado localmente, es decir, será entregado a un proceso que se ejecuta localmente en el equipo.

- Si la dirección destino del paquete es distinta de alguna dirección local del equipo, se evalúa la cadena FORWARD, sí y sólo sí, el equipo local tiene activo el reenvío de paquetes (de forma predeterminada está desactivado). Si se superan las reglas de la cadena FORWARD se reenvía el paquete por la interfaz correspondiente, pero si la función de reenvío no estaba habilitada, el paquete se descarta (DROP).

El segundo caso ocurre cuando los procesos locales generan paquetes, estos entran en Netfilter por la cadena OUTPUT y los paquetes que pasen por la cadena OUTPUT necesariamente pasan por POSTROUTING. Así, en todos los casos, antes de que los paquetes abandonen el sistema Netfilter el último gancho es POSTROUTING y si se superan todas las cadenas de este gancho, serán enviados a la interfaz destino. Para comprender mejor el funcionamiento se muestran a continuación de forma esquemática el recorrido que pueden realizar los paquetes en función de su destino.

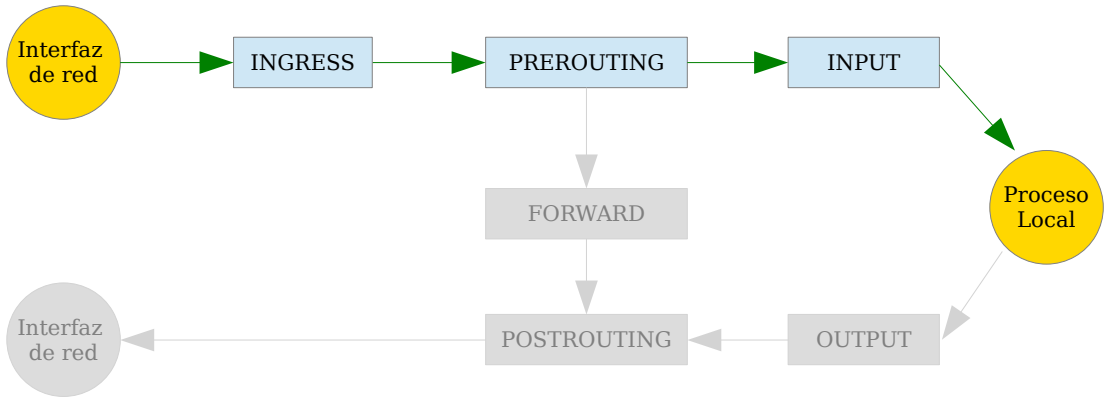


Figura 2. Flujo de paquetes entrantes con destino local.

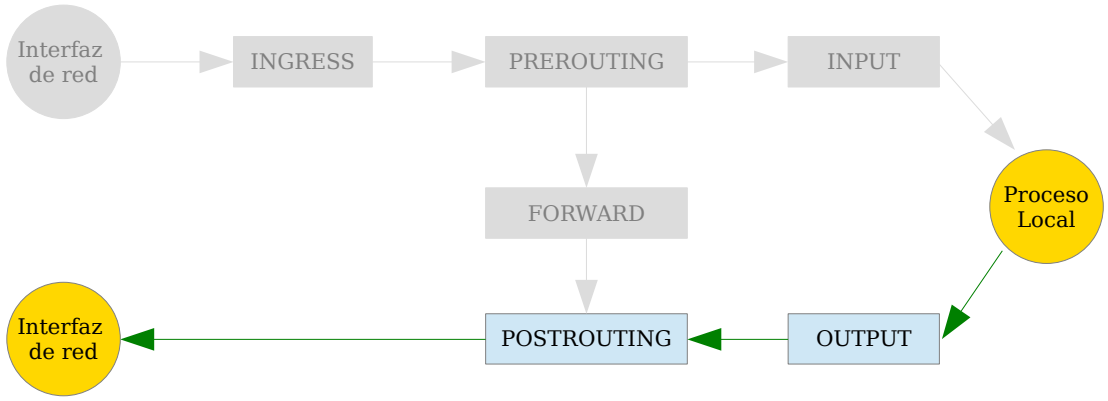


Figura 3. Flujo de paquetes salientes originados en local.

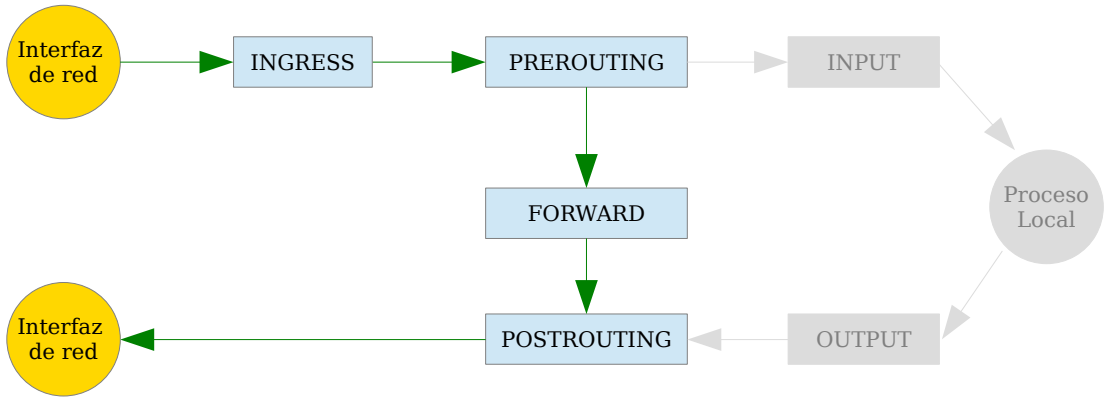


Figura 4. Flujo de paquetes reenviados.

3.3. Cadenas en Netfilter con NFT

Tras las tablas y los ganchos, el siguiente componente de Netfilter son las cadenas de reglas. Las cadenas son agrupaciones de reglas que se aplican a los paquetes en momentos concretos (ganchos) del flujo de los mismos en su paso por el sistema Netfilter. De este modo, las cadenas suelen estar enlazadas a algún gancho, aunque en configuraciones avanzadas se pueden crear cadenas sin gancho asociado. Las reglas contenidas en una cadena constituyen una lista ordenada y son evaluadas de una en una empezando por la primera que se especifica y no siempre se evalúan todas, ya que una regla puede realizar un veredicto (acción final) sobre el paquete terminando el procesamiento en esa regla.

Las cadenas sólo pueden ser definidas dentro de una tabla y las tablas son obligatoriamente de una determinada familia, este hecho determina el tipo de paquetes que se pueden procesar en las cadenas y el tipo de reglas admitidas. Esto es así ya que, como se explicó anteriormente, cuando se crea una tabla hay que indicar a la familia de direcciones con la que trabaja: *ip*, *arp*, *netdev*, etc. Las cadenas se definen dentro de una tabla siguiendo la siguiente sintaxis:

```
chain <nombre_cadena> {
    type <tipo> hook <gancho> priority <prioridad>
    <lista_de_reglas>
}
```

En la sintaxis mostrada, al especificar una cadena hay que indicar el tipo de procesamiento que realizará en dicha cadena, el gancho al que quedará enlazada y la prioridad. Una cadena sólo puede ser de un tipo y quedar enlazada a un único gancho, consiguiendo así que todos los paquetes que fluyen por ese punto de procesamiento se puedan analizar y procesar en la cadena.

Para poder trabajar con las cadenas es necesario comprender en que consiste el *tipo* de cadena. Éste hace referencia a las posibles acciones que se pueden realizar con los paquetes, y los tipos pueden ser: *filter*, *route* o *nat*. Tras establecer el tipo de cadena, en las reglas se podrán ejecutar ciertas acciones condicionales sobre los paquetes, pero sólo serán válidas aquellas acciones permitidas por el tipo de cadena, por ejemplo, si la cadena es de tipo *filter* se podrá aceptar o descartar el paquete cuando cumpla una regla. En cambio, si la

cadena es de tipo *nat*, no es posible descartar paquetes. Estas acciones,algunas de ellas llamadas sentencias o veredictos, se estudiarán posteriormente en el documento.

Volviendo a la especificación de la cadena, ésta se complica cuando se quiere enlazar a un gancho, ya que el gancho no puede ser cualquiera, está restringido al tipo de cadena, y aún más, el tipo de cadena está restringido a la familia de la tabla que contiene la cadena. Para tratar con todas estas restricciones se presentan las posibles combinaciones en la tabla 1. Para entenderlo hay que recordar de nuevo que, las tablas están asociadas a una familia de direcciones la cual se especifica al crear la tabla. Dentro de la tabla están las cadenas, y el tipo de la cadena no pueden ser utilizado arbitrariamente con cualquier tabla. Existe una triple limitación, para las combinaciones de familia de tabla, tipo de cadena y ganchos disponibles, el cual se resume la siguiente tabla.

Familia tabla	**Tipo de Cadena**	**Ganchos**
arp	filter	input, output
ip / ipv6 / inet	filter	prerouting, input, forward, output, postrouting
	route	output
	nat	prerouting, input, output, postrouting
bridge	filter	input, forward, output
netdev	filter	ingress

Tabla 1. Restricciones en las tablas, tipos y ganchos disponibles para las cadenas.

También hay que considerar que nftables permite crear cadenas sin enlazar a ningún gancho, pensadas para ser usados en firewalls complejos donde se necesiten ordenar y agrupar reglas, pero no se contemplan este tipo de cadenas en los ejemplos tratados en este documento. Como punto de partida en el ejemplo siguiente se muestran dos cadenas sin reglas, pero con los ganchos especificados.

```
#!/usr/sbin/nft -f

flush ruleset

table inet filtrado_ip {
    chain prueba {
        type filter hook postrouting priority 200;
    }
}

table arp filtrado_mac {
    chain prueba {
        type filter hook input priority 100;
    }
}
```

Código 2. Contenido de un fichero con un ejemplo de creación de tablas y cadenas.

Respecto a los tipos de cadenas, se resume a continuación su objetivo ya que cada tipo de cadena permitirá realizar determinadas acciones con los paquetes. Las acciones se estudian en profundidad a lo largo de lo que queda de documento junto con las reglas y mediante ejemplos. Los tipos principales son los siguientes:

- FILTER: Se usa para el filtrado general de paquetes y principalmente permite aceptar o descartar paquetes.
- NAT: Controla la traducción de direcciones. Permite alterar las direcciones origen y destino del datagrama. Algunos usos típicos de NAT son:
  - SNAT (Source NAT): También conocido como *IP Masquerading*. Se cambia la dirección IP origen del datagrama después del encaminamiento y antes de su reenvío.
  - DNAT (Destination NAT): Se modifica la dirección IP destino antes del encaminamiento. Algunas aplicaciones de DNAT son:
    - Proxy transparente: Se redirecciona el datagrama a otro equipo que será quien proporcione los servicios requeridos.
    - Balanceo de carga: Se cambia la dirección destino de los datos recibidos en un balanceador para redirigirlos a una máquina concreta en función del servicio requerido o entre máquinas distintas para balancear la carga.
    - Port Forwarding: El router recibe las peticiones para la subred en la que trabaja y cambia la IP hacia la que tiene que dirigirse.
- ROUTE: Analiza ciertas características del paquete y lo marca en función de su naturaleza para que reciba cierto tratamiento específico (ej: diferenciar tráfico en función de los servicios, etc.).

Otro aspecto importante es la llamada *política predeterminada* de una cadena. Las cadenas además del tipo y el gancho tienen una política por defecto y hace referencia a la acción a realizar cuando un paquete no cumple ninguna de las reglas de la cadena. Se puede considerar como la sentencia o acción final cuando se han evaluado todas las reglas de la cadena. Existen dos posibles políticas para las cadenas: *accept* (aceptar) o *drop* (descartar) y en caso de no indicarse explícitamente, la política predeterminada es *accept*.

La finalidad al establecer una política predeterminada es crear un cortafuegos de tipo permisivo (*accept*) o prohibitivo (*drop*). Resulta fundamental definir bien cuál es la política por defecto más conveniente. Si lo que se desea es un sistema lo más restringido posible (prohibitivo), entonces lo más conveniente es descartar cualquier tipo de tráfico excepto el que se autorice explícitamente. En este caso podemos comenzar impidiendo cualquier tráfico saliente para después añadir tan sólo aquellas comunicaciones que deseamos autorizar, como por ejemplo, los accesos al servidor DNS y las conexiones SSH a un determinado servidor. Cualquier otro tráfico distinto del autorizado será rechazado por esa restrictiva política predeterminada.

Sin embargo, también es posible que lo que deseemos sea tan sólo impedir cierto tipo de tráfico sin alterar el resto de servicios. Quizá queremos evitar que una determinada aplicación pueda funcionar, por ejemplo, que los usuarios no puedan imprimir en una cierta impresora remota desde ese ordenador. En este caso se impone partir de una política por defecto que acepte todo tipo de tráfico (cortafuegos permisivo), para después introducir una regla que bloquee específicamente el tráfico que se dirija a esa impresora de red.

Con nftables la sintaxis para establecer la política es sencilla, basta con añadir una línea con el contenido `policy accept` o `policy drop`. En el ejemplo mostrado a continuación se ha establecido una política prohibitiva para la cadena *filtro_salida*. Dicha cadena está enlazada al gancho de salida y no contiene ninguna regla, por lo que si prueba dicho ejemplo dejará sin tráfico saliente IP al equipo.

```
#!/usr/sbin/nft -f

flush ruleset

table ip filtrado_ip {
    chain filtro_salida {
        type filter hook output priority 0;
        policy drop;
    }
}
```

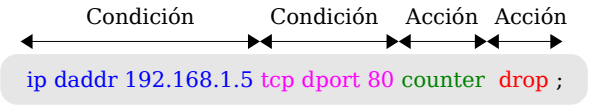
Código 3. Ejemplo de política de salida en el gancho de salida.

3.4. Reglas y acciones

Para terminar de definir una cadena hay que añadir un listado de reglas a la cadena. Como se indicó anteriormente, las cadenas están formadas por una lista ordenada de reglas y éstas serán evaluadas en el orden en el que se especifiquen en la cadena. La finalidad de una regla es realizar alguna acción sobre un paquete en caso de cumplirse todas sus condiciones. Así, cada regla consiste en una o varias condiciones, que, en caso de cumplirse todas ejecutan una o varias acciones. Cuando una regla se cumple y se disparan las acciones no siempre se sigue procesando la cadena por la siguiente regla. Este comportamiento queda definido por el tipo de acción ejecutada tras cumplirse las condiciones de una regla: si alguna de las acciones es una acción de las denominadas *acción final*, se detiene el procesamiento de la cadena en esta regla. Si todas las acciones son *no final* se continúa por la siguiente regla.

El formato general de una regla es una concatenación de condiciones seguida de una concatenación de acciones: REGLA := <condiciones> <acciones>. Todas las acciones de una regla se ejecutan sólo si todas las condiciones se cumplen, es decir, las condiciones están concatenadas de forma implícita por el operador lógico AND. Existen multitud de condiciones en Netfilter y cada una de ellas consiste en una comparación con algún valor de los que maneja internamente Netfilter. Al ser tantos, en este documento sólo se presentan los más relevantes. Para comenzar a estudiar las reglas se parte de un ejemplo con una regla con 2 condiciones y 2 acciones. Esta regla comprueba una determinada dirección IP de destino y un determinado puerto TCP de destino y contiene dos acciones, una de las cuales, es final (*drop*) consistente en descartar el paquete. En caso de cumplirse esta regla, el paquete se descarta y no se procesa ninguna más y su sintaxis sería:

En este primer ejemplo hay dos condiciones y en ambas se ha hecho la comparación de un elemento de la cabecera de un paquete (campo IP en la primera condición, campo TCP en la segunda condición). El operador usado es "==" y puede ser omitido, de forma que, si en una condición no aparece un operador lógico se aplica el operador *igual* "==". Lo habitual es omitir este comparador quedando la regla anterior equivalente a:



Además del operador igual se pueden usar otros operadores y otros tipos de expresiones como son rangos, máscaras, etc. En la siguiente tabla se presentan algunos ejemplos usando como base las 2 condiciones presentadas en la regla anterior.

Operador / Selector	Descripción / Ejemplos
==	Operador predeterminado, puede omitirse. `Ejemplo:` `tcp dport == 443`
!=	`Ejemplo:` `tcp dport != 80`
gt ó >	`Ejemplo:` `tcp dport > 1024`
lt ó <	`Ejemplo:` `tcp dport < 1024`
<= y >=	`Ejemplos:` `tcp dport <= 1024` `tcp dport >= 1000`

<inicio> - <fin>	Selector de rangos válido tanto para direcciones IP como para valores numéricos.
	`Ejemplos:` `ip daddr 192.168.1.100 – 192.168.1.200`
	`tcp dport 6000 - 7000`

<ip>/<mascara>	Selector de máscara de bits `Ejemplo:` `ip daddr 192.168.1.0/24`

{<elementos> }	Conjunto de elementos separado por coma: direcciones IPs, valores numéricos, etc.
	`Ejemplo:` `ip daddr {192.168.1.5, 192.167.1.6 }`
	`tcp dport {22, 80, 443 }`

Tabla 2. Operadores y selectores admitidos en las condiciones de las reglas.

Además de las condiciones hay que saber manejar correctamente las acciones, las cuales se disparan cuando en una regla se cumplen todas las condiciones. Se distinguen dos tipos de acciones a*cciones no finales* y *acciones finales*, también llamados éstas últimas veredictos. Una primera restricción es que una regla con varias acciones sólo puede tener una única acción final. Se debe a que las acciones finales en una regla indican el destino de un paquete o de una transacción, de este modo, al dispararse se termina el procesado de la cadena. A continuación se presentan las acciones finales básicas:

Acción final	Descripción
accept	Aceptar el paquete/transacción de la cadena. No asegura que el paquete es aceptado, ya que podrá ser evaluado por otras cadenas.
drop	Descartar el paquete/transacción. Se descarta de forma inmediata y no se traspasa a ninguna otra cadena.
reject	Envía un paquete con una respuesta de error. Sólo es válida en para los ganchos: *input*,*forward* y *output*.

continue	Continua la evaluación de la siguiente regla, la acción implícita cuando no se especifica ninguna.
jump <cadena>	Salta a la primera regla de otra cadena, es posible volver a esta cadena mediante un *return* desde la cadena a la que se ha saltado.
return	Vuelve a la cadena que ejecutó un jump hacia esta cadena. Continúa en la siguiente regla tras el jump.
goto <cadena>	Salta a otra cadena pero sin posibilidad de volver mediante return.

Tabla 3. Acciones finales más relevantes.

Existen multitud de acciones no finales en la documentación de *nftables*, por ello sólo se trataran las más comunes. Algunas de las presentadas a continuación tienen gran cantidad de opciones y no se presentan todas las posibilidades.

Acción no final	**Descripción / Ejemplos**
counter	Añade un contador del número de veces que se dispara la regla. El contador se ve listando las reglas.
log [prefix] [level] [flags]	Hace que el núcleo escriba información en la bitácora cuando se dispare la regla. Existen varias opciones sobre la forma de escribir mensajes, se recomienda leer documentación detallada antes de usarla. `Ejemplo:` `log level info`
limit rate <unidades/seg>	Añade un limitador de trafico del tipo cubeta con fichas.
<expresión> set <valor>	Permite alterar el contenido de un paquete. `Ejemplo:` `ip ttl set 5`
meta <prop> set <valor>	Establece el valor de una propiedad interna de Netfilter a un valor determinado. Pueden ser: ▪ mark: Marca arbitraria para el usuario. ▪ nftrace: Depurador/monitor de reglas. ▪ pktype: host, broadcast, multicas, other. ▪ piority: relacionado con el sistema conformado de trafico (traffic shapper, tratado en otro documento). `Ejemplo:` `meta mark set 15`

Tabla 4. Acciones no finales más relevantes.

Una vez presentadas las acciones básicas se va a detallar el algoritmo de procesado de las cadenas. Si se parte de una cadena con varias reglas y una política predeterminada, sólo se procesarán aquellos paquetes correspondientes al gancho. Para cada paquete que entre en la cadena el algoritmo es el siguiente:

1. Se toma la primera regla de todas las reglas existentes en la cadena.
2. Se evalúan las condiciones de la regla:

2.1. Si no se cumplen todas las condiciones se continua a la siguiente regla.

2.2. Si se cumplen todas las condiciones, se ejecutan las acciones asociadas:

2.2.1. Si hay una acción final, se detiene el procesamiento de la cadena.

2.2.2. Si no hay ninguna acción es final se, continua con la siguiente regla de la cadena.

3. Si se terminan las reglas de la cadena y no se ha cumplido la condición de ninguna regla, se ejecuta la política por defecto sobre el paquete (por ejemplo, *accept* ó *drop*).

En el procesamiento descrito es importante el orden en el que están las reglas, fíjese en en el siguiente ejemplo, donde se intentan contar los paquetes dirigidos a una IP y además bloquear el puerto 80. Este ejemplo muestra un error común cuando no se considera el orden:

```
chain entrada {
  type filter hook input priority 0;
  ip daddr 192.168.1.5 tcp dport 80 drop;
  ip daddr 192.168.1.5 counter;
}
```

```
chain entrada {
  type filter hook input priority 0;
  ip daddr 192.168.1.5 counter;
  ip daddr 192.168.1.5 tcp dport 80 drop;
}
```

Código 4. Ejemplo de influencia en el orden de las reglas.

En el ejemplo anterior la acción final *drop* detiene la evaluación de las reglas de la cadena, por ello en la solución de la izquierda el contador se incrementa para todos los paquetes dirigidos a la IP 192.168.1.5 menos los dirigidos al puerto 80. En cambio en la solución de la derecha la regla que tiene el contador siempre se evalúa y se pasa a la siguiente regla, donde se descartan los paquetes dirigidos al puerto 80.

Tras describir el funcionamiento de las cadenas se profundizará en la elaboración de reglas. Como ya se ha explicado, las reglas se construyen por concatenación de condiciones y acciones asociadas. Cada condición evalúa una propiedad del paquete. Para indicar a Netfilter que hacer con los paquetes de una transacción, se deben crear reglas lo más precisas posible. La idea es que la condición sea inequívoca, tanto como para quien creó la regla (usuario) como para el núcleo. Una condición (coincidencia / match) ocurre cuando un paquete cumple con los criterios indicados en la regla. Alguno de estos criterios pueden basarse en algún parámetro como, el tipo de protocolo (TCP, IP, ICMP, etc.), algún puerto en particular, un usuario propietario/generador del paquete (OWNER), el estado de la transacción (INVALID), o la combinación de todos ellos. Las condiciones se construyen usando una serie de operadores que determinan la lógica que el paquete debe cumplir. Con *nftables* las condiciones se pueden asociar a tres grupos de condiciones diferentes, según los datos que se comprueban:

1. Aquellas que comprueba el contenido de los paquetes (*payload*) en diferentes niveles de red .
2. Aquellas que comprueban los metadatos (*meta data*) asociados al paquete como son interfaces de red, proceso que lo generó, etc .
3. Aquellas capaces de comprobar el estado de las máquinas de estado (*stateful*) que gobiernan los protocolos dentro del núcleo, (por ejemplo, el protocolo TCP).
4. A continuación se presentan las condiciones y acciones las más relevantes para cada uno de los

grupos, pero este documento no se tratarán todas las condiciones y acciones posibles, en [https://www.netfilter.org/projects/nftables] están detalladas todas las posibilidades.

3.4.1. Condiciones para comprobación del contenido de paquetes

Las condiciones más habituales son aquellas en las que se examinan el contenido de las cabeceras de los paquetes de red. Netfilter contempla el análisis de las cabeceras de los paquetes desde diferentes niveles de red: *arp*, *ipv4*, *ipv6*, *icmp*, *tcp*, *udp*, etc. El más básico y el más utilizado es el análisis de las cabeceras en el nivel IP y en el siguiente nivel con los protocolos más habituales como TCP y UDP. En este sentido, se muestra a continuación una figura con los campos de la cabecera en un datagrama IP, para después en una tabla, presentar la sintaxis usada en nftables analizar dichos los campos.

Figura 5. Descripción a nivel de bits de la cabecera de un datagrama IP.

Condición	Descripción

ip protocol <protocolo>	Esta condición es cierta si el campo protocolo del paquete IP coincide. El protocolo corresponde con un valor numérico de 8 bits, aunque admite las cadenas *tcp*, *udp*, *icmp*, *etc.* con los nombres del protocolo.
	`Ejemplos:` `ip protocol tcp`
	`ip protocol {udp,icmp}`

ip saddr <ip>	Comprueba la dirección IP del origen del paquete. Puede indicarse también la máscara de la forma IP/máscara para decirle a Netfilter que se trata de un grupo de hosts o red. `Ejemplo:` `ip saddr 192.168.1.0/24`
ip daddr <ip>	Comprueba la dirección IP del destino del paquete. Puede indicarse también la máscara de la forma IP/máscara para decirle a Netfilter que se trata de un grupo de hosts o red. `Ejemplo:` `ip daddr {192.168.1.5/32, 192.168.1.6/32}`
ip ttl <ttl>	Comprobación del campo TTL del paquete. Puede ser útil combinar el valor de 8 bits con un operador mayor o menor. `Ejemplo:` `ip ttl > 5`
ip length <length>	Comprobación del tamaño el paquete

Tabla 5. Condiciones de comprobación de la cabecera IP.

Además de estas condiciones existen otras, que junto a éstas, extienden sus funcionalidades concretando aún más la condición que se desea determinar. Para muchos de los protocolos IP existen comprobaciones directas para las los campos de sus cabeceras, como por ejemplo para TCP, UDP e ICMP, entre otros. En estos casos no es necesaria anteponer en la regla la condición `ip protocol tcp`, basta con usar una condición `tcp`.

Fíjese en el siguiente ejemplo que añade un contador para los paquetes dirigidos a un servidor Web (HTTP puerto 80):

```
ip protocol tcp tcp dport 80 counter;
        1           2          3
```

Esta regla tiene dos condiciones que deben cumplirse: (1) protocolo IP==6 (TCP) y (2) el puerto destino del protocolo TCP==80. Pero la condición (1) queda implícita en la segunda, ya que el protocolo TCP es un protocolo transportado por el protocolo IP, por ello la regla se puede simplificar mediante:

```
tcp dport 80 counter;
      1          2
```

De igual modo que con el protocolo IP, en la siguiente figura se ha añadido y resaltado de color azul la cabecera TCP y los campos, de un datagrama TCP/IP.

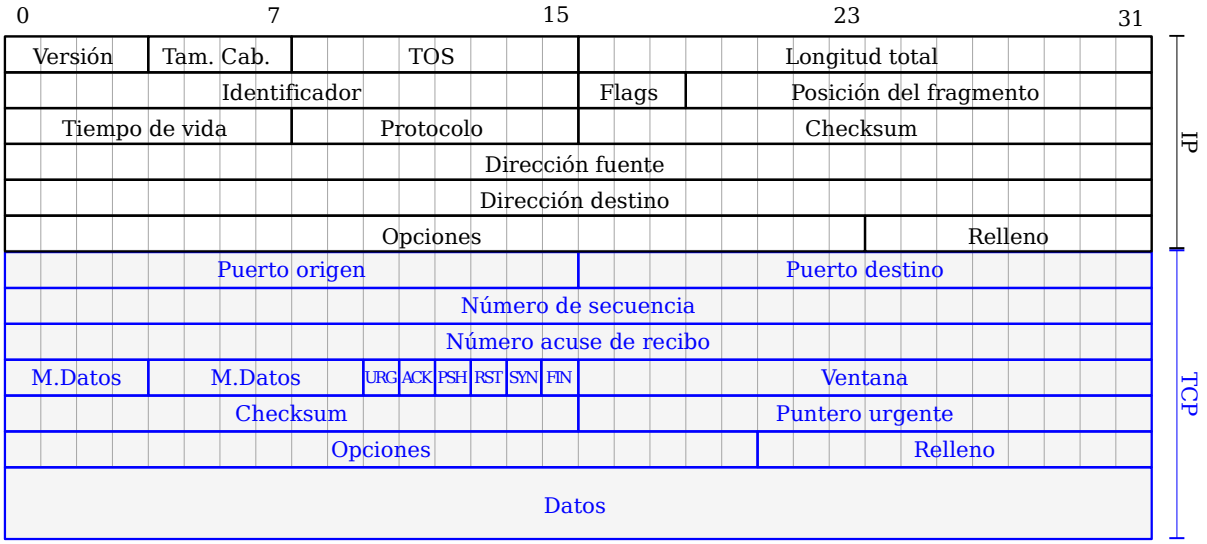


Figura 6. Descripción a nivel de bits de la cabecera de un datagrama TCP-IP.

Considerando paquetes TCP/IP, UDP/IP e ICMP/IP en las siguientes tablas se resumen algunas de las comprobaciones que se pueden hacer en estas cabeceras incluyendo algunos ejemplos de las mismas.

Condición TCP	Descripción

tcp sport <puerto>	Comprobación del puerto origen.
	`Ejemplos:` `tcp sport 53`
	`tcp sport 20-30`

tcp dport <puerto>	Comprobación del puerto destino.
	`Ejemplos:` `tcp dport 23`
	`tcp dport {80,443}`

tcp flags <flags>	Para los paquetes TCP se comprueban una serie de indicadores. Se debe añadir un argumento que se puede ser un indicador, varios indicadores o una expresión. El parámetro *flags* es una máscara de bits que admite los indicadores: *fin*, *syn*, *rst*, *psh*, *ack*, *urg*, *ecn*, *cwr*.
	▪ La máscara se puede formar separando por coma los flags a testar
	▪ De forma avanzada la máscara se puede formar usando varios flags con los operadores a nivel de bit "&"/and y "|"/or, realizando con el resultado una comparación binaria.
	`Ejemplos:` `tcp flags ack,syn`
	`tcp flags != ack`
	`tcp flags & (syn | ack) == (syn)`

tcp window <valor>	Comprobación del tamaño de recepción indicado en el paquete `Ejemplo:` `tcp window < 12`
tcp ackseq <valor>	Número de secuencia del paquete cuyo valor es de 32bits.

Tabla 6. Condiciones de comprobación de cabeceras del protocolo TCP.

Condición UDP	Descripción
udp sport <puerto>	Condición de puerto origen. `Ejemplo:` `udp sport 0-1023`
udp dport <puerto>	Condición de puerto destino. `Ejemplo:` `udp dport { 6666, 6668 }`
length <longitud>	Comprobación de la longitud de los datos, número de 16 bits `Ejemplo:` `udp length < 10`

Tabla 7. Condiciones de comprobación de cabeceras del protocolo UDP.

Condición ICMP	Descripción

icmp type <tipo>	Comprueba el tipo de paquetes ICMP. Se puede poner un número de bits o listar los predefinidos con el comando `nft describe icmp type`.
	`Ejemplos:` `icmp type echo-reply`
	`icmp type time-exceded`

icmp mtu <valor>	Condición para comprobar el campo MTU (valor de 16 bits) `Ejemplo:` `icmp mtu == 1492`
icmp sequence <valor>	`Condición para comprobar el número de secuencia (valor de 16 bits)` `Ejemplo:` `icmp sequence > 20`

Tabla 8. Condiciones de comprobación de cabeceras del protocolo ICMP.

3.4.2. Condiciones para comprobación de metadatos de paquetes

Además de poder analizar el contenido de los paquetes, Netfilter añade una estructura de datos interna y adicional con información sobre el paquete, la cual, contiene propiedades como puede ser la interfaz por la que entró el paquete. Algunas de estas propiedades son establecidas por Netfilter pero otras pueden cambiarse durante el procesado mediante reglas. Esto permite marcar paquetes, es decir, establecer una propiedad a un valor durante el tiempo que el paquete es procesado dentro del núcleo de Linux. Esta información adicional anexada a cada paquete de red se llama meta-información y en la siguiente tabla se muestran las propiedades más utilizadas.

Condición META	Descripción

meta iif <interfaz>	Comprueba la interfaz de entrada, desde donde se realiza la transacción o se reciben los paquetes, para una regla en particular.
	`Ejemplos:` `meta iif eth0`
	`meta iif != lo`

meta oif <interfaz>	Comprueba la interfaz de salida de la transacción o paquete
	`Ejemplos:` `meta oif eth1`
	`meta oif {eth0,eth1}`

meta iiftype <tipo>	Comprueba el tipo de interfaz de entrada, es útil para distinguir ethernet, túneles u otros. Para obtener los tipos predefinidos se puede usar el comando `nft describe meta iiftype`. `Ejemplo:` `meta iiftype != ether`
meta oiftype <tipo>	Comprueba el tipo de interfaz de salida del mismo modo que *iiftype*. `Ejemplo:` `meta oiftype ppp`
meta length <length>	Longitud del paquete en bytes.

meta mark [set] <marca>	Sirve para comprobar la marca de un paquete o para establecer la marca a un valor. La marca es un valor arbitrario que el usuario puede asociar a un paquete para usarlo para cualquier fin.
	`Ejemplos:` `meta mark 12`
	`meta mark != 23`
	`meta mark set 0x55`

meta skuid <uid>	Comprueba el UID (User ID) del usuario del sistema que originó el paquete. `Ejemplo:` `meta skuid < 1000`
meta skgid <uid>	Comprueba el GID (Group ID) del usuario del sistema que originó el paquete.
meta pkttype	Comprueba el tipo de paquete de entre los siguientes: *broadcast, unicast, multicast, other*. `Ejemplo:` `meta pkttype != broadcast`

Tabla 9. Condiciones de comprobación de los metadatos.

3.4.3. Seguimiento de conexiones (connection tracking).

Las máquinas de estado implementadas en el núcleo de Linux para manejar los protocolos son accesibles desde Netfilter mediante el módulo llamado *conntrack*. El uso de este módulo es complejo y sólo es

necesario en configuraciones avanzadas. A continuación sólo se muestran las 4 comprobaciones más comunes.

Condición conntrack	Descripción
ct state <estado>	Comprueba el estado de la conexión para un paquete. Puede ser: ▪ new: Creación de nueva conexión. ▪ established: El paquete forma parte de una conexión establecida. ▪ related: Conexión relacionada a una ya establecida. ▪ untracked: Paquetes no identificados en ninguna conexión. ▪ invalid: Paquetes no válidos. `Ejemplo:` `ct state != established`
ct status <estado>	Comprueba el tipo de conexión pudiendo ser estado: *expected*, *seen-reply*, *assured*, *confirmed*, *snat*, *dnat*, *dying*. `Ejemplo:` `ct status seen-reply`
ct direction <reply|original>	Dirección de la conexión `Ejemplo:` `ct direction reply`

ct mark [set]	Permite marcar conexiones para su procesamiento. Se pueden realizar operaciones a nivel de bits con los operadores: *or*, *and* y *xor*.
	`Ejemplo:` `ct mark 0xff`
	`ct mark and 0x80 == 0x80`
	`ct mark set 0xf0`

Tabla 10. Condiciones de comprobación en el seguimiento de conexiones.

3.5. Network Addresss Translation (NAT)

El módulo NAT con Netfilter requiere el uso cadenas específicas de tipo *nat*. Estas cadenas sólo son válidas en tablas de la familia *ip* y tienen acciones específicas para realizar las diferentes operaciones NAT. Estas operaciones son esencialmente *source NAT* (SNAT) y *destination NAT* (DNAT) aunque según el modo en que se realicen, aparecen en Netfilter dos más: enmascaramiento (*masquerade*) y redirección (*redirect*).

El enmascaramiento (*masquerade*) es un tipo de SNAT donde la dirección se cambia la dirección fuente del paquete a una dirección no determinada a priori. Cuando se determina la ruta del paquete y la interfaz por la que debe salir el paquetes, se usa la dirección IP de salida de esa ruta para sustituir la dirección IP fuente original del paquete. Por otro lado, la redirección (*redirect*) es un tipo específico de DNAT cuyo objetivo es redirigir de forma transparente el tráfico a la máquina local, por ejemplo, de un puerto TCP a otro. La primera restricción de NAT es que sólo pueden realizarse acciones NAT con las tablas de familia IP y la segunda es que, cada una de las acciones disponibles con NAT sólo es posible en determinados ganchos. Estas restricciones se muestran en la tabla 11.

Acción	Ganchos
snat	input, postrouting
dnat	output, prerouting
masquerade	postrouting
redirect	output, prerouting

Tabla 11. Ganchos válidos para las acciones NAT.

Considerando las restricciones de la tabla 11, en el ejemplo mostrado en el código 5 se presentan dos cadenas, la primera con una regla para redirigir las conexiones entrantes al puerto TCP443 hacia el puerto 8006, y la segunda, permite compartir Internet del mismo modo que lo hacen los routers convencionales. En el ejemplo se supone que la IP 80.22.51.2 es la dirección pública y 192.168.0.0/24 la red privada.

```
#!/usr/sbin/nft -f

flush ruleset

table ip probando_nat {

    chain probando_dnat {
        type nat hook prerouting priority 0;
        policy accept;
        ip daddr 80.22.51.2/32 tcp dport 443 dnat :8006
    }

    chain probando_snat {
        type nat hook postrouting priority 100;
        policy accept;
        ip saddr 192.168.0.0/24 snat 80.22.51.2
    }
}
```

Código 5. Ejemplo de cadenas con acciones NAT.

En la siguiente tabla se muestran las acciones NAT disponibles y su uso ya que muchas de ellas requieren parámetros.

Acción	Descripción / Ejemplos
dnat <IP> [:puerto/s]	Usa *destination NAT* consistente en cambiar la dirección/puerto destino por los indicados. Sólo se puede usar con los ganchos *prerouting* y *output*. `Ejemplo:` `dnat 192.168.1.12:8080`
snat <IP> [:puerto/s]	Cambia la dirección IP de origen a la especificada usando *Source NAT*. Sólo se puede aplicar en los ganchos *postrouting* e *input*. `Ejemplo:` `snat 11.22.33.44`
masquerade [tipo] [:puerto/s]	Cambia la dirección IP de origen a usada en la ruta por defecto de la red. Sólo se puede usar en el gancho *postrouting* y admite algunas opciones extra para el algoritmo de asignación de puertos TCP: *persistent*, *random* o *fully-random*. `Ejemplo:` `masquerade random`

Acción	Descripción / Ejemplos
redirect to [:puerto/s]	Traslada la dirección a una local. Sólo se puede usar con los ganchos *prerouting* y *output*.

Tabla 12. Acciones NAT.

3.6. Opciones avanzadas

La utilidad *nftables* incluye un modo adicional para monitorización y depuración del firewall. Tiene dos modos de funcionamiento, el primero, permite monitorizar en tiempo real las transacciones/eventos del conjunto de reglas, mostrando las reglas que se añaden o borran. El segundo modo, permite marcar determinadas reglas y obtener información cuando se disparan. En ambos modos el procedimiento consiste en ejecutar el comando `nft monitor` con los parámetros adecuados en un terminal, de forma que, el terminal muestra por la salida estándar los eventos de monitorización hasta terminar el comando de forma manual (CTRL+C o kill). En ta tabla 13 se muestran los dos modos y los parámetros usados.

Comando	Descripción / Ejemplos
nft monitor [evento] [filtro]	Monitoriza todos los eventos de Netfilter. Se puede filtrar por tipo de evento o tipo de elemento: ▪ filtro: sólo muestra los cambios producidos en un determinado elemento, siendo válidos: *tables*, *chains*, *sets*, *rules*, *elements* y *ruleset*. ▪ filtro: permite mostrar sólo los elementos añadidos o los elementos eliminados mediante las opciones: *new* o *destroy*.
nft monitor trace	Permite la depuración sólo de las reglas marcadas previamente con `nftrace set 1`.

Tabla 13. Opciones de monitorización con nftables.

Para monitorizar determinadas reglas es necesario marcar dicha regla mediante la acción no final `nftrace set 1`. Una vez cargadas las reglas en Netfilter, el monitoreo se realiza usando el comando `nft monitor trace` en un terminal. En el código 6 se muestra un ejemplo para monitorizar mediante una condición del módulo *conntrack* las nuevas conexiones entrantes.

```
table inet curioseando {
    chain entrada {
        type filter hook input priority 0;
        ct state new nftrace set 1;
    }
}
```

Código 6. Ejemplo de regla con monitoreo activo.

Este tipo de monitorización necesita tener un terminal o un proceso en segundo plano con el comando `monitor trace` en ejecución. Alternativamente existe otro modo para usar el registro de sucesos del sistema (bitácora) para registrar el disparo de las reglas. Para este propósito, existe una acción *log* no final que cuando se dispara, entre otras opciones, puede escribir un mensaje en el registro de sucesos. El registro de

sucesos del sistema se puede ver mediante el fichero /var/log/syslog o el comando dmesg . La acción *log* en su formato más básico admite tres parámetros opcionales siendo su sintaxis la siguiente:

```
log [prefix texto_descriptivo] [level nivel_de_log] [flags datos_a_mostrar]
```

Para comprenderlo mejor, fíjese en el siguiente ejemplo con dos reglas, las cuales, sólo registran los inicios de conexión TCP a dos puertos determinados (SSH y OpenVPN), dejando un mensaje en el registro de sucesos. En la segunda regla se escribe un mensaje de texto adicional descriptivo para facilitar la lectura.

```
table inet cortafuegos {
    chain entrada {
        type filter hook input priority 0;
        ct state new tcp dport 22 log;
        ct state new tcp dport 1194 log prefix "Conexion entrante OpenVPN: ";
    }
}
```

Código 7. Ejemplo de reglas con escritura en el registro de sucesos.

Existen más opciones en Netfilter/nftables aparte de las mostradas en este manual y documentadas tanto en la página de manual del comando nft como en la página oficial de Netfilter. Pero antes de finalizar este manual, se añade un ejemplo útil para firewalls complejos donde es recomendable usar una sintaxis legible que facilite el mantenimiento. En el ejemplo mostrado en el código 8 se presentan dos elementos nuevos de sintaxis: (1) la posibilidad de dividir la configuración en varios ficheros, para después unirlos mediante la sentencia *include* y (2) la definición de constantes (*define*) para la reutilización en diferentes reglas. También se muestra como el carácter # permite escribir comentarios.

```
#!/usr/sbin/nft -f

flush ruleset

# Ejemplo de inclusión de ficheros externos
include "/etc/nftables/redlocal.nft"
include "/etc/nftables/servicios-externos.nft"

# Ejemplos de definiciones
define ip_servidor_web = 192.168.0.7
define puertos_web = {80,443}
define servidores_dns = { 1.1.1.1, 8.8.8.8, 8.8.4.4 }

table inet cortafuegos {
    chain entrada {
        type filter hook input priority 0;
        policy drop;
        ip saddr $servidores_dns accept;  # Para usar una definición se precede de '$'
    }
    chain salida {
        type filter hook output priority 0;
        policy drop;
        ip daddr $servidores_dns accept;
    }
    chain reenvio {
        type filter hook forward priority 0;
        policy drop;
```

```
        ip daddr $ip_servidor_web tcp dport $puertos_web counter accept;
        ip saddr $ip_servidor_web tcp sport $puertos_web accept;
    }
}
```

Código 8. Ejemplo con definiciones e inclusión de ficheros.

4. Realización del laboratorio

La realización de este laboratorio está dividida en cuatro bloques. El primero consistirá en convertir la máquina GW en un router para que de acceso a Internet a las máquina de la red privada. Después se harán ejemplos guiados para ir probando el efecto de diferentes reglas del firewall en los equipos. La tercera parte es no guiada, donde cada estudiante debe elaborar una regla NFT para el firewall dadas unas especificaciones. El último bloque consiste en un caso práctico de configuración de firewall que deberá estar operativo durante los restantes laboratorios.

En el primer ejercicio se plantea a continuación y además de conseguir compartir Internet se instalará un escritorio en la máquina VM3 para usarla como puesto de trabajo en la red privada. En importante que todas la pruebas realizadas a lo largo del laboratorio se guarden en diferentes ficheros en la ruta */root/firewall* ya que posteriormente serán objeto de evaluación, se irá indicando el nombre de cada fichero en cada tarea.

Tarea 1.- Entre en el panel de control de las máquinas virtuales de la asignatura e inicie la máquina GW pero no abra la consola desde la Web, debe hacer este ejercicio conectando por SSH de manera remota.

T1.1.- Compruebe que la red pública de su máquina GW está configurada correctamente. Para ello desde el equipo local, abra un terminal y use el comando `ping 192.168.20.X`, siendo X la dirección asignada en el curso.

T1.2.- Use CTRL+C para parar el comando *ping* y entre en su máquina por SSH usando el comando `ssh tai@192.168.20.X` (de nuevo X debe ser su dirección).

T1.3.- Abra otro terminal local en otra ventana y repita el proceso para tener 2 conexiones simultáneas a su máquina GW y trabajar de forma más fluida.

T1.4.- Conviértase en administrador en los dos terminales mediante el comando `sudo su`.

En primer lugar se propone crear un script ejecutable que contenga las reglas necesarias para que la máquina GW comparta la conexión a Internet usando *Source Nat* (SNAT). Esta primera configuración no tendrá ningún tipo de restricción y será útil para poder cargarla en caso de problemas con el firewall. En la figura 7 se muestra un ejemplo de una conexión saliente a la que se le aplica SNAT al pasar por el router. Básicamente consiste en que el router cambia la dirección origen de la paquete TCP por su IP pública y para cada respuesta a esa conexión, se cambia dirección destino hacia la máquina que inició la conexión.

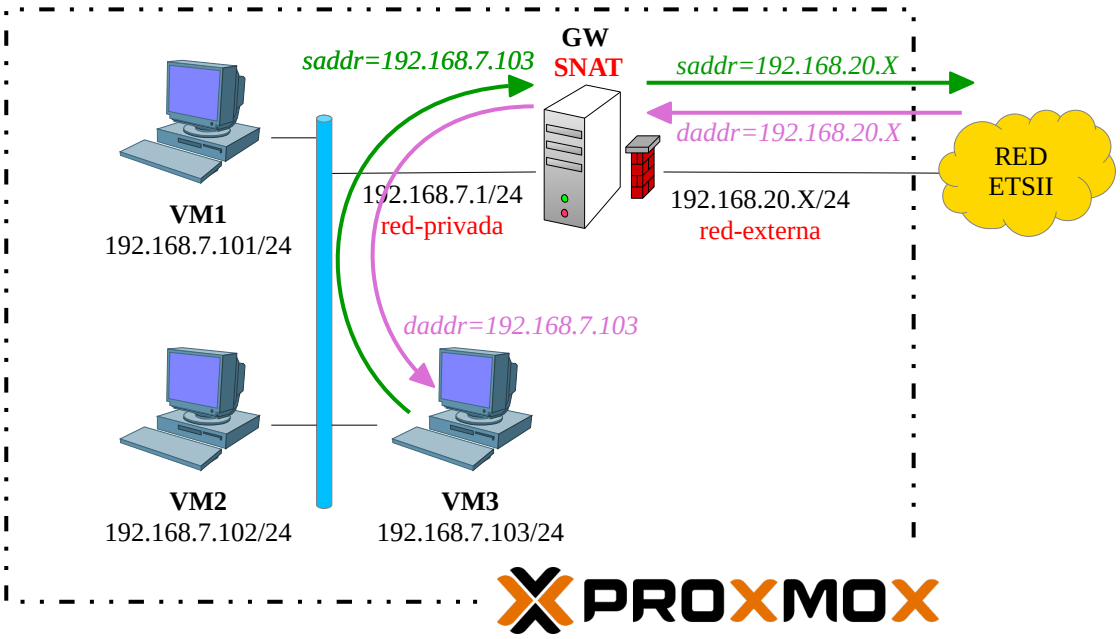


Figura 7. Ejemplo de conexión SNAT.

Tarea 2.- Cree un nuevo directorio `/root/firewall` y entre en él usando el comando `cd`. Asegúrese estar en el directorio correcto usando el comando `pwd`.

T2.1.- Cree un nuevo fichero en el directorio */root/firewall* llamado *nat-permisivo.nft*. Compruebe con `pwd` que no se ha equivocado de directorio y establezca el permiso de ejecución al fichero. Edítelo y en la primera línea escriba `#!/usr/sbin/nft -f`.

```
#!/usr/sbin/nft -f

flush ruleset
```

T2.2.- Debe completar el fichero anterior usando partes del código 5 con el objetivo de disponer de una regla que permita el reenvío de paquetes a través del router, desde la red 192.168.7.0/24. Así se compartirá la conexión a Internet mediante SNAT. Para conseguirlo haga lo siguiente:

- Del código 5 mostrado, borre sólo la cadena DNAT y mantenga la cadena SNAT.
- Adapte la cadena SNAT sabiendo que su red privada es la 192.168.7.0/24 y su IP pública es 192.168.20.X.

T2.3.- Para cargar este fichero existen 2 opciones: (1) usar el comando `nft -f nat-permisivo.nft`, (2) convertir en ejecutable el fichero mediante el comando `chmod +x nat-permisivo.nft`. En este curso se optará por la segunda opción. Usando el comando indicado convierta en ejecutable el fichero `nat-permisivo.nft` y compruebe que efectivamente es ejecutable, listando el directorio con `ls -l`. ¿Observa la una X en el listado de permisos del fichero? ¿Sabe interpretar la salida del comando `ls -l`? ¿Por qué la primera línea del fichero de texto tiene `#!/usr/sbin/nft -f`? ¿que relación tiene esto con `chmod +x`?

T2.4.- Para poder ejecutar un fichero, desde el directorio actual de trabajo se debe escribir en la línea de

comandos el nombre del fichero precedido de ./, por ejemplo para ejecutar el fichero anterior hay que escribir `./nat-permisivo.nft`. Ejecute este fichero para cargarlo en memoria mediante `./nat-permisivo.nft`, comprobando que no contiene errores. También debe ejecutar `nft list ruleset` para ver si lo cargado en memoria coincide con el contenido del fichero.

T2.5.- Para que opere la regla anterior, debe activar el reenvío de paquetes del núcleo del sistema operativo. Los sistemas operativos en general nunca tienen activa esta opción de forma predeterminada. Debe editar el fichero */etc/sysctl.conf* y activar la línea `net.ipv4.ip_forward=1`. Ejecute el comando `sysctl -p` para que se aplique la nueva configuración.

T2.6.- Compruebe si las máquinas *VM1, VM2 y VM3* tienen acceso a Internet usando el comando *ping* hacia alguna dirección IP externa, no use resolución de nombres ya que no está configurada todavía.

T2.7.- En caso de operar correctamente, configure los DNS de VM1, VM2 y VM3. Para ello, compare el fichero /etc/network/interfaces de GW con el mismo fichero en VM1, VM2 y VM3 y haga las modificaciones oportunas. Tras ello se recomienda reiniciarlas máquinas y utilizar el comando ping o host con un nombre de dominio existente, por ejemplo `ping google.es` o `host www.us.es`, verificando que el sistema resuelve a la IP correspondiente.

Este último fichero debe guardarse durante todo el curso por contener configuración de filtrado totalmente permisivo. Cuando se trabaje posteriormente en el firewall prohibitivo, en caso de problemas se puede volver a cargar esta configuración para realizar comprobaciones.

Ahora se propone realizar una instalación mínima para un escritorio gráfico. La instalación de un escritorio no es necesaria para la administración y aumenta el tamaño de la imagen en más de 600MBytes, lo cual aumenta el tiempo y tamaño de exportación pero se utilizará para facilitar el desarrollo de los laboratorios. Sea consciente que, en administración de sistemas, nunca se instala un escritorio en un servidor.

Existen multitud de escritorios disponibles en las diferentes distribuciones de GNU-Linux, siendo los de menor peso y que consumen pocos recursos *lx*, *xfce* y *mate*. En principio, no basta con instalar algún escritorio, se necesita un entorno gráfico capaz de manejar los controladores de vídeo siendo el más usado en Linux XORG. En este curso se propone hacer una mínima instalación del escritorio *mate* junto con un gestor de sesiones llamado *lightdm*. El sistema de dependencias de Debian detectará automáticamente los paquetes adicionales necesarios para que estos dos funcionen, los instalará y los configurará automáticamente.

Tarea 3.- Revise en el panel de control de Proxmox la memoria disponible en cada máquina. Debe asegurarse de que la máquina VM3 dispone de más de 1GB de RAM para poder instalar un escritorio.

T3.1.- Antes de instalar software adicional es siempre necesario actualizar la base de datos de los paquetes disponibles en la red y descargar las últimas actualizaciones. Los comandos mostrados a continuación debe usarlos en la máquina VM3 y actualizan la lista de paquetes, aplican las actualizaciones e instalan el escritorio *mate*:

```
apt update
apt upgrade
apt install mate lightdm
apt autoremove
```

Código 9. Uso del comando apt para instalar software.

T3.2.- Una vez terminada la instalación reinicie la máquina VM3 usando el comando `reboot` como administrador. Deberá aparecer una pantalla de bienvenida en modo gráfico en la que podrá iniciar una sesión de escritorio.

T3.3.- Entre en el menú de aplicaciones, acceda al menú Sistema → Centro de control y aumente la resolución de pantalla al menos a 1024x768, aunque es recomendable disponer de una mayor resolución a lo largo del curso.

Para poder trabajar de forma cómoda con las máquinas virtuales es recomendable habilitar el soporte de portapapeles compartido. Esto permite seleccionar textos en el equipo local y pegarlos en el escritorio remoto, y viceversa. Aunque no es necesario para la realización de los laboratorios, esta característica facilitará mucho el trabajo puesto que, a lo largo de los laboratorios tendrá que copiar fragmentos grandes de texto en las máquinas virtuales. En la siguiente tarea se propone instalar y testar el soporte compartido de portapapeles el cual funciona tanto en equipos locales Linux como MsWindows.

Tarea 4.- En el menú de aplicaciones del escritorio recién instalado inicie el emulador de terminal, lo encontrará en herramientas del sistema. Usando el terminal cambie a la cuenta de administrador mediante el comando `sudo su`.

T4.1.- Intente copiar con el ratón de este PDF el comando `apt install spice-vdagent` y pegarlo en el terminal de la máquina virtual desde el menú Editar → Pegar. Observará que no funcionará, para poder usar esta característica debe teclear el comando anterior para instalar el soporte de compartir el portapapeles.

T4.2.- Reinicie la máquina, abra el editor de textos que está en Accesorios → Pluma de la máquina virtual y vuelva a intentar copiar y pegar desde el equipo local a la máquina virtual. Puede probar el editor de texto Pluma de la máquina virtual para seleccionar un texto en su equipo local y pegarlo en la máquina virtual. Debería funcionar correctamente.

T4.3.- Instale el gestor de paquetes con entorno gráfico llamado *Synaptic*: `apt install synaptic`.

T4.4.- Busque en los menús del escritorio el Gestor de paquetes Synaptic y ejecútelo. Usando este programa use el buscador incorporado para localizar el navegador Web Firefox e instálelo, no se confunda de programa ya que hay muchas opciones, lea la descripción para ver que es cada paquete.

T4.5.- Una vez instalado, busque en el menú de programas del escritorio *Firefox* y ejecútelo para asegurarse que ha instalado el paquete correcto.

Para poder realizar el resto de laboratorio se usará la máquina GW que hace de router de una red privada a la que está conectada la máquina VM3 con escritorio. Este escenario es similar al existente en cualquier instalación básica con Internet.

4.1. Ejercicios guiados.

La primera parte del laboratorio consiste en realizar un ejercicio guiado mostrando los comandos

necesarios para cada una de las tareas planteadas. Recuerde que todas las pruebas realizadas a lo largo del ejercicio se irán guardando en diferentes ficheros en la ruta */root/firewall* de la máquina GW y que posteriormente serán objeto de evaluación.

Tarea 5.- Conecte a la máquina GW mediante SSH, conviértase en administrador y entre en el directorio */root/firewall*. Una vez dentro, realice las reglas mostradas en el fichero indicado en cada tarea.

T5.1.- Cree en este directorio un nuevo fichero llamado fw1.nft, para ello basta con usar el editor con el nuevo fichero como argumento: `nano fw1.nft`. Antes de trabajar en este fichero debe añadir en las primeras líneas lo indicado a continuación:

```
#!/usr/sbin/nft -f

flush ruleset

include "./nat-permisivo.nft"
```

T5.2.- Tiene que añadir permiso de ejecución a este fichero (revise T2.3.- y T2.4.-) y, ejecutarlo desde el directorio actual del modo `./fw1.nft`. Una vez ejecutado, debe comprobar si se incluye correctamente el fichero *nat-permisivo.nft*, para ello, ejecute `nft list ruleset` y revise que aparecen las reglas SNAT.

T5.3.- Añada en el fichero anterior la tabla con la cadena del ejemplo mostrado en el código 3 (pág. 11), el cual contenía una política de descarte (prohibitiva) para el gancho de salida. Debe guardar el fichero de texto y volver a la línea de comandos.

T5.4.- Ejecute este script de firewall y si todo ha ido bien, habrá perdido la conexión con el servidor por SSH en todas las terminales. ¿Por qué ha ocurrido esto?.

T5.5.- Para poder solucionarlo la única forma es entrar desde el panel de control Web del servidor y abrir una consola desde el navegador para borrar todas las reglas del firewall. Debe ejecutar el comando `nft flush ruleset` como administrador para recuperar la comunicación exterior con el servidor.

En el ejemplo usado, la orden policy drop que modifica la política por defecto para todo el tráfico saliente, incluido aquel que no abandona el sistema (*localhost*), de modo que, que ningún paquete puede salir por ninguna interfaz. El efecto es parecido a desconectar los cables de red.

Tarea 6.- Trabajando sobre el mismo fichero, fw1.nft, se deben añadir y comprobar el correcto funcionamiento varias reglas, trabajando siempre sobre la única cadena del fichero.

T6.1.- Cambie la política a accept y cargue el fichero de nuevo para comprobar que no ha perdido la conexión.

T6.2.- En la única cadena existente se va a añadir una regla que descarte en la salida los paquetes con el protocolo IP-ICMP. Añada a la cadena la regla `ip protocol icmp drop;` y recargue las reglas desde el fichero. Para comprobar si la regla está filtrando tráfico, use el comando `ping` hacia su dirección IP pública desde otro equipo de la red.

T6.3.- NFT contiene acciones no finales como counter. Ésta añade un contador con el número de veces que se dispara una regla. No se debe utilizar en modo de producción ya que reduce el rendimiento pero es útil para la depuración y pruebas del firewall. Para probarlo modifique la regla que descarta los

paquetes ICMP: `ip protocol icmp counter drop;`. Recargue de nuevo el firewall y compruebe que se siguen filtrando los paquetes ICMP con el comando `ping`.

T6.4.- Para ver el valor del contador sólo es necesario listar en cualquier momento el conjunto de reglas con el comando `nft list ruleset` cuya salida es:

```
table ip filtrado_ip {
        chain filtro_salida {
                type filter hook output priority 0; policy accept;
                ip protocol icmp counter packets 23 bytes 1932 drop;
        }
}
```

Muchas infraestructuras sitúan un cortafuegos entre su red local y la conexión a Internet. Este dispositivo incluye algunas reglas que filtran determinados paquetes con el fin de mejorar la seguridad de la red privada. En este laboratorio se puede hacer que nuestro equipo descarte, de manera selectiva determinado tipo de tráfico. Para ello vamos a necesitar reglas un poco más elaboradas que la empleada en el ejercicio anterior.

Tarea 7.- Ahora se va a trabajar sobre un nuevo fichero llamado fw2.nft, para no comenzar desde cero, puede copiar el anterior usando el comando `cp fw1.nft fw2.nft`. En el nuevo fichero debe añadir dos nuevas cadenas de reglas, una en el gancho de entrada y otra en el gancho de reenvío. Es decir, en el fichero habrá 3 cadenas con el contenido mostrado a continuación (fíjese en los ganchos/hooks):

```
#!/usr/sbin/nft -f

flush ruleset

include "./nat-permisivo.nft"

table ip filtrado_ip {

    chain filtro_salida {
        type filter hook output priority 0;

    }
    chain filtro_entrada {
        type filter hook input priority 0;

    }
    chain reenvio {
        type filter hook forward priority 0;

    }
}
```

T7.1.- Se desea bloquear el tráfico existente en la interfaz privada (red interna VM1,VM2 y VM3). Use el comando `ip addr` para listar los dispositivos y revise en la salida cual el es el nombre de la interfaz con la IP 192.168.7.1, ¿eth0 o eth1?. Después, añada en la cadena de entrada la regla de filtrado: `meta iif eth? counter drop;` (sustituya la ?) y cargue el fichero mediante `./fw2.nft`.

T7.2.- Para realizar la prueba debe usar dos terminales, el primero en VM3 ejecutando `ping 192.168.7.1`. Éste último comando se quedará bloqueado, y en GW desde otro terminal debe ejecutar varias veces el comando `nft list ruleset` para observar como se incrementa el contador de la regla.

T7.3.- Para no tener que ejecutar reiteradamente el comando anterior, existe una utilidad que ejecuta reiteradamente un comando cada 2 segundos llamada `watch`, tras la cual se escribe el comando que se desea ejecutar indefinidamente. Para probarla ejecute en el terminal libre el siguiente comando: `watch nft list ruleset`, verá que el terminal ejecuta el comando pero queda también bloqueado. Para desbloquear ambos terminales utilice CTRL+C.

T7.4.- Desde VM3 compruebe que sólo ha bloqueado la red privada haciendo *ping* a alguna dirección externa, por ejemplo `ping www.dte.us.es` o `ping 8.8.8.8`.

En este ejercicio se ha visto como mediante la condición *meta iif* se especifica un dispositivo de red, el cual no es parte del paquete IP. En el ejemplo se ha establecido que todo el tráfico que se reciba (*input*) por el dispositivo conectado a la red privada debe descartarse (*drop*), pero en cambiom VM3 sigue teniendo acceso la red exterior. Para conseguir reglas más útiles no basta poder especificar el dispositivo, sino que es necesario poder afinar indicando qué protocolo y/o puertos disparan una regla en particular.

Tarea 8.- En esta tarea también debe tener abierta una conexión SSH con su máquina GW como administrador. Se va a trabajar sobre un nuevo fichero fw3.nft pero partiendo del anterior (fw2.nft) así que copie el fichero fw2.nft como fw3.nft. y borre todas las reglas. En el nuevo fichero añada en la cadena asociada al gancho output (salida) la regla: `tcp sport 22 counter drop;`

T8.1.- Cargue el firewall con `./fw3.nft` e intente seguir trabajando mediante la conexión SSH y teclee algún comando. ¿Qué sucede? ¿Por qué?.

T8.2.- Para desactivar el firewall es necesario ejecutar el comando `nft flush ruleset`, pero no puede hacerlo mediante SSH porque está bloqueado. Debe entrar desde la Web del servidor de virtualización y abrir una consola para ejecutar el comando anterior como administrador.

T8.3.- Una vez eliminado el firewall de memoria ¿Qué sucede? ¿Aún sigue conectado por SSH al equipo remoto?

T8.4.- Edite el fichero fw3.nft y comente (no la borre) la regla problemática anteponiendo el carácter # a la regla. Ahora piense en una regla que tenga el mismo efecto pero usando la cadena asociada al gancho *input*. Una vez añadida, verifique que funciona.

T8.5.- Si la regla anterior está bien hecha, desde VM3 no puede conectarse por SSH a GW (comando: `ssh 192.168.7.1`). En cambio, compruebe que sí puede conectar por SSH a cualquier otra máquina exterior, por ejemplo, usando su *uvus* pruebe: `ssh uvus@192.168.20.3`. Debe evitar que se reenvíen paquetes SSH a la red exterior, así que cree una nueva regla en la cadena *forward* que bloque SSH desde la red privada a todo Internet. Verifique que se han cerrado las conexiones salientes SSH hacia el exterior desde VM1, VM2 y VM3.

En este ejemplo se ha creado una regla que no especifica el dispositivo de red sino el protocolo (TCP) y también el puerto origen (*sport 22*). En una conexión SSH los paquetes que salen del servidor SSH tienen como puerto origen el 22 que es el puerto predeterminado del servicio SSH. De modo análogo es posible aplicar esta misma estrategia para bloquear el acceso a cualquier otro servicio. No obstante, las reglas se pueden aplicar tanto al tráfico entrante como al saliente, o bien a ambos. En el ejercicio anterior se bloqueaba el tráfico SSH saliente (*output*). Si se tarda más de un minuto entre el paso 1 y el paso 2 del ejercicio, es

posible que la conexión SSH se haya interrumpido. Si la conexión se interrumpió lo más conveniente es repetir el ejercicio intentando ser algo más rápido (lo que permitirá que la conexión no se interrumpa y se obtenga un resultado diferente).

Por otro lado, un error muy común a la hora de configurar el corta fuegos es poner las reglas en las cadenas *input* y *output*, pensando que corresponden a los paquetes que entran y salen por las interfaces de red. Para no cometer este error hay que revisar la figura 4 que hace referencia a los paquetes entrantes pero cuyo destino no es la máquina GW, denominados paquetes reenviados y que sólo pasan por forward. La cadena *input* y *output* es sólo para los paquetes cuyo destino/origen sea la propia máquina GW, es decir, hacia o desde las IPS: 192.168.7.1 y 192.168.20.X (siendo X exactamente la IP asignada a cada alumno).

En los siguientes ejemplos se van a crear reglas que atiendan a las direcciones de los paquetes, combinados con los puertos. En la elaboración de reglas con direcciones IP y puertos TCP/UDP es importante considerar el sentido en el viaja el paquete, siendo este último aspecto origen de confusión y errores en la elaboración de reglas. En este sentido, considere que cuando especifique una regla los paquetes IP tienen una dirección de origen y una de destino, al igual que los puertos, un puerto origen y uno destino, por lo que aparecen 4 posibles combinaciones, de las cuales, según el cometido de la regla, sólo una o dos serán necesarias. Además, todos los ejemplos serán para conexiones salientes desde la red privada, por ello, esos paquetes sólo pasan por *forward*. En los siguientes ejemplos se intenta profundizar en este aspecto.

Tarea 9.- Se trabajará sobre un nuevo fichero llamado fw4.nft, puede copiar el fichero fw3.nft con el comando `cp fw3.nft fw4.nft` y después borrar todas las reglas para dejar las cadenas de entrada, salida y reenvío vacías. En esta tarea, se necesitará conectar al escritorio de la máquina VM3, debe conectar un escritorio remoto por la Web de Proxmox, no por SSH. El objetivo es bloquear páginas WEB.

T9.1.- Una vez que esté en el escritorio remoto de VM3, abra un navegador y cargue la página *http://www.dte.us.es*.

T9.2.- Ahora añada una regla para el gancho de reenvío (*forward*) de la siguiente forma: `ip daddr www.dte.us.es tcp dport 443 counter drop;`. Cargue el nuevo firewall comprobando que no hay errores.

T9.3.- Vuelva al navegador Web y pulse F12 (herramientas de desarrollador) y seleccione en las herramientas del navegador el panel de red (network). Con el panel abierto recargue la página en el navegador con F5. ¿Qué ocurre? ¿Está bloqueando el tráfico HTTP o HTTPS?

T9.4.- Pruebe a navegar de nuevo la página mediante la dirección *https://www.dte.us.es*. Debería estar bloqueada.

T9.5.- Para bloquear un sitio Web completamente es necesario es necesario bloquear dos protocolos HTTP y HTTPS (versión insegura y la segura), la solución sería añadir otra regla para el puerto 80 (HTTP), y aquí nftables permite crear conjuntos de elementos como puertos o direcciones IP. Modifique la regla de forma que quede como sigue:

```
ip daddr www.dte.us.es tcp dport {80,443} counter drop;
```

T9.6.- Recargue las modificaciones del firewall y vuelva a probar a navegar la página Web del

Departamento. Verifique que puede visitar cualquier otra página sin problemas.

Se puede observar en este ejercicio como se ha creado una regla que descarta el tráfico TCP hacia al servidor *www.dte.us.es* y destinado al los puertos 443 y 80 (HTTPS/HTTP). Esta regla no afecta al tráfico similar enviado a cualquier otro servidor. Con esta regla sólo se impide el acceso al servidor Web principal del DTE. Es posible crear un conjunto de reglas similares para poder impedir el acceso a una lista de distintos servidores Web.

4.2. Seguimiento de reglas y registro de sucesos

La funcionalidad de nftables no sólo permite especificar reglas para descartar paquetes como hemos visto en varios de los ejemplos. Estableciendo una política prohibitiva (*drop*) que descarte paquetes, son necesarias más reglas, y deberán ser especificadas para aceptar determinados tipos de tráfico. Otra funcionalidad interesante es que las reglas pueden producir acciones de registro, que se anotarán en el registro de sucesos del sistema (archivo */var/log/syslog*). Para crear reglas que al dispararse generen una entrada en el registro se usará la acción *log*. Esta acción es *no final* por lo que no determinan si el paquete se acepta o se rechaza, de forma que se aplicará la acción final que corresponda como si esta regla de registro no existiera.

El interés de poder registrar determinados sucesos asociados con el tráfico de red depende de las situaciones que se estén considerando. Puede tener un efecto informativo para el administrador sobre multitud de datos que pudieran estar registrados en otros lugares o no. Por ejemplo, si deseamos conocer cuántas personas se conectan cada día a nuestro servidor FTP es muy posible que el programa servidor disponga de un detallado archivo de registro con esa información, pero si se trata de un servidor muy elemental podría no generar tipo alguno de información de registro.

Además de esta posibilidad de usar el registro, nftables incluye otro modo muy útil para depuración o inspección del firewall en tiempo real llamado modo monitor. Éste modo es capaz de mostrar los eventos de Netfilter como son la creación o borrado de tablas y reglas. En este modo también es posible marcar determinadas reglas para que, en caso de dispararse, se muestre información en el modo monitor.

A continuación se realizará un ejemplo presentado los dos modos y suponiendo que se desea determinar cuántos clientes se conectan cada día a un servidor mediante el protocolo SSH. Lo primero que necesitamos es determinar que condición consideramos como válida para establecer que ha llegado un nuevo cliente. La más sencilla (aunque no necesariamente la más correcta) es considerar que cada nueva conexión al puerto 22 de nuestro servidor es un nuevo cliente.

Tarea 10.- Se trabajará sobre un nuevo fichero `fw5.nft`. Puede copiar uno de los anteriores con el comando `cp` y editarlo, dejando sin reglas todas las cadenas. Esta tarea debe realizarse usando dos terminales conectados simultáneamente a la máquina GW, o alternativamente un terminal usando el programa `tmux`. [Tmux Quick Reference].

T10.1.- En un primer terminal ejecute el comando `nft monitor` y verá como se queda el terminal en espera. En un segundo terminal cargue el fichero `fw5.nft` recién creado, no importa que no contenga reglas. ¿Observa cambios en el terminal que monitoriza nftables?.

T10.2.- El protocolo SSH opera en TCP en el puerto 22. Para monitorizar esta conexión se propone

detectar los paquetes entrantes TCP hacia este puerto. Edite el fichero fw5.nft y en la cadena enlazada al gancho input añada la siguiente regla: `tcp dport 22 nftrace set 1;`. ¿Por qué en este caso se usa *input* y no *forward*?

T10.3.- Cargue de nuevo el fichero fw5.nft para que surjan efecto los cambios y tras la carga, observe el terminal donde tiene activa la monitorización. ¿Por qué no para de mostrar información de manera continuada?. Detenga la monitorización con las teclas CTRL+C para no colapsar el terminal.

T10.4.- El objetivo es sólo monitorizar los inicios de conexión, que serán paquetes TCP entrantes con la bandera SYN activa. Sustituya la regla anterior por la regla: `tcp dport 22 tcp flags syn nftrace set 1;`. Recargue las nuevas reglas desde el fichero fw5.nft.

T10.5.- Inicie de nuevo la monitorización en otro terminal con `nft monitor`. ¿Se muestra algún evento en la monitorización?. Pruebe desde un tercer terminal iniciar una sesión SSH para ver si aparecen datos de monitorización.

Si se analiza el ejercicio realizado se puede concluir lo útil que es disponer del sistema de monitorización para probar determinadas reglas antes establecerlas de forma definitiva. Con la primera regla ocurrió que cada vez que un paquete llega al equipo y va destinado al servidor SSH se disparaba la regla. El problema reside en que esa regla se cumple para cada segmento SSH recibido de cada conexión, eso quiere decir que un mismo cliente puede generar miles de disparos de la regla en una misma conexión, ejecutando miles de veces la acción asociada y ralentizando el sistema. En la segunda regla se refinó más la condición de forma que, sólo se detectaran los paquetes de inicio de conexión disminuyendo el número de disparos de la regla.

Tarea 11.- Una vez conseguida una regla que sólo muestra ciertos eventos, se desea que no sea necesario usar la monitorización interactiva, sino que se graben entradas en el registro del sistema para que sean persistentes usando la acción log.

T11.1.- Debe añadir una nueva regla en el fichero fw5.nft igual que la mostrada en T10.4.- pero cambiando la acción, de forma quedará: `tcp dport 22 tcp flags syn nftrace set 1 log;`. Tras añadirla recargue el firewall desde el fichero y realice una nueva conexión SSH desde el ordenador local para conseguir que se dispare.

T11.2.- Para verificar si está operando correctamente es necesario visualizar el contenido del fichero /var/log/syslog. Existen múltiples formas de ver este fichero en una consola, pero se recomienda usar el programa less que permite avanzar/retroceder por el fichero y mostrar los cambios en tiempo real. Ejecute el comando `less /var/log/syslog` y analice las últimas líneas que aparecen, puede avanzar y retroceder con las teclas: *AvPag* / *RePag* / *Fin* / *Inicio*. ¿Encuentra alguna línea con información del firewall?

T11.3.- Ahora pulse la combinación de teclas SHIFT+F y se quedará el terminal a la espera de nuevas líneas en el fichero. Inicie y cierre varias veces una conexión SSH desde el equipo local observando el terminal que está ejecutando *less*.

T11.4.- Para desbloquear el terminal y salir del programa *less* debe pulsar la combinación de teclas CTRL+C y después la tecla q.

4.3. Modificando direcciones y/o puertos de destino (NAT)

Otra de las funciones ampliamente usada en los firewall son las diferentes modalidades NAT. Se propone modificar direcciones y/o puertos de destino NAT, en el siguiente ejercicio creando una regla en la que se aplica *Destination NAT* (DNAT).

Tarea 12.- Para realizar este ejercicio es necesario que conecte al escritorio remoto de la máquina VM3. En VM3, abra el navegador Firefox e inicie una ventana de navegación anónima. Visite la página http://www.dte.us.es. teclee HTTP no HTTPS.

T12.1.- Cree un nuevo fichero fw6.nft con el contenido mostrado (no se olvide usar el comando `chmod +x` para añadir permiso de ejecución y poder cargarlo posteriormente):

```
#!/usr/sbin/nft -f

flush ruleset
include "./nat-permisivo.nft"

table ip cortafuegos {
    chain nat_salida {
        type nat hook prerouting priority 0;
        ip daddr 150.214.141.196 tcp dport 80 dnat to 150.214.188.143:80
    }
}
```

T12.2.- Mediante el uso del comando `ping` o el comando `host` averigüe el nombre asociado la dirección IP 150.214.188.143 del sitio Web destino. Cargue las reglas del fichero fw6.nft y en el escritorio de VM3 recargue en Firefox la página http://www.dte.us.es pulsado la tecla F5. ¿Qué página ve ahora sin haber cambiado la dirección?

T12.3.- Repita la tarea anterior añadiendo una regla similar, pero para el protocolo HTTPS (puerto 443). Recargue el firewall y navegue a https://www.dte.us.es. ¿Por qué con el protocolo HTTP Firefox no detecta una posible alteración del sitio Web con HTTP y con HTTPS sí?

T12.4.- Elimine todas las reglas ejecutando el comando `nft flush ruleset` para evitar problemas.

Si repasamos la configuración de nftables mostradas, presentan algunas diferencias respecto a las utilizadas en la sección anterior. En primer lugar, la tabla que alberga la cadena es de la familia *ip* no *inet* como anteriormente. El motivo de esto es que NAT debe operar sobre direcciones IPv4 y las tablas *inet* son para IPv4 e IPv6. La segunda, es que la cadena es de tipo *nat (*`type nat`*)*, ya que, para aplicar acciones NAT es necesario este tipo de cadenas, (*filter* no dispone de acciones NAT). Finalmente la regla permite reescribir las direcciones y/o los puertos de destino de un segmento. En este caso todos los segmentos TCP dirigidos a la dirección IP de la WEB del DTE (150.214.141.196) serán redirigidos al servidor web de la dirección 150.214.188.143 (portal del LSI), evidentemente solo para puertos 80 y 443.

Esta nueva tabla permite realizar operaciones de traducción de puertos y direcciones como las que realiza un router convencional. Con reglas DNAT no sólo es posible redirigir las conexiones salientes, sino que también las conexiones entrantes TCP/UDP hacia otras direcciones IP, este tipo de reglas corresponde al mecanismo mal-llamado *abrir puertos* en los routers convencionales, el cual, simplemente es DNAT.

4.4. Cambios en otros campos

Otra de las posibilidades existentes con Netfilter es la modificación de los valores de campos en el tráfico analizado. Uno de los posibles es el campo DSCP de la cabecera IP asociado a la calidad de servicio (QoS), por ejemplo, para realizar tareas de marcado. Incluso aunque su valor no sea respetado más allá de nuestro sistema, puede ser interesante para que distintos flujos de tráfico simultáneos se puedan tratar adecuadamente según nuestra elección, en caso de tener una red de cierta complejidad.

Para el último ejemplo vamos a escoger algo más sencillo, se propone alterar el campo de tiempo de vida (TTL) de la cabecera IP. El TTL especifica el número máximo de saltos entre routers permitidos para alcanzar un destino. El valor utilizado por nuestro equipo para este campo viene prefijado en la configuración del sistema operativo y aunque se puede modificar, la mayoría de administradores no suelen tener razones para hacerlo. Sin embargo, no todos los sistemas operativos utilizan el mismo valor, y si nos fijamos, es posible que veamos computadores con distintos valores iniciales para el TTL, se puede usar el comando *ping* para determinarlo. Supongamos que deseamos modificar el valor del campo TTL para cierto tipo de tráfico (no para todos nuestros paquetes). Es posible crear una regla con Netfilter que realice ese cambio selectivo.

Tarea 13.- En GW use el comando `traceroute` hacia 193.147.175.38 (*www.us.es*) para averiguar los saltos hasta llegar a la web de la Universidad. Este comando no siempre funciona bien debido a que muchos routers intermedios no son amigables y bloquean los paquetes ICMP por motivos de seguridad. De hecho, pruebe `ping` hacia `www.informatica.us.es` y observará que no hay respuesta.

T13.1.- Instale la herramienta *mtr* con el comando `apt install mtr` y úsela para el caso anterior mediante el comando `mtr www.us.es` y compárelo con `mtr www.informatica.us.es`. Si está en una consola, para parar esta herramienta debe usar CTRL+C, no basta con cerrar la ventana.

T13.2.- Ejecute de nuevo los comandos anteriores pero con la opción `-n` es decir `mtr -n destino`. Observará que no se resuelven nombres y se muestran las IPs de cada salto ¿Detrás de cuantos niveles de NAT está su conexión? ¿Cual es su IP pública de salida a Internet?

T13.3.- Realice la misma prueba con *www.google.es* y con los servidores DNS públicos 8.8.8.8 y 1.1.1.1. ¿De quién son esos DNS públicos y cual de ellos funcionará mejor interpretando los datos en tiempo real de MTR?

T13.4.- Ahora se va a añadir una configuración de firewall con una regla que altere el valor TTL de los paquetes salientes a 4. Para hacerlo, cree un nuevo fichero `fw7.nft` con el contenido mostrado y no olvide añadir permiso de ejecución para cargarlo posteriormente.

```
#!/usr/sbin/nft -f
flush ruleset
include "./nat-permisivo.nft"

table inet cortafuegos {
    chain post_rutado {
        type filter hook postrouting priority 0;
        oif eth0 ip ttl set 4;
    }
}
```

T13.5.- Acceda al escritorio remoto de VM3 e intente navegar usando Firefox a www.dte.us.es, www.google.es. y a algún otro sitio exterior de la red de la Universidad.

T13.6.- Vuelva a usar la herramienta mtr con los objetivos anteriores (T13.3.-) y observará un comportamiento anómalo.

T13.7.- Finalmente elimine todas las tablas con el comando `nft flush ruleset` para evitar futuros problemas.

Lo que ha observado es como usando ese valor tan sólo se pueden realizar cuatro saltos antes de que el paquete IP sea descartado. Esto limita enormemente el número de redes que se pueden visitar. Un valor TTL=1 impediría atravesar un único router y solo permitiría la comunicación directa con otros ordenadores en la misma red.

4.5. Ejercicios no guidados

El objetivo de las tareas de esta sección es crear reglas y probarlas mediante un test que debe cubrir el objetivo deseado. Para realizar las comprobaciones se utilizará la herramienta *netcat*, por eso, la siguiente tarea presenta el uso básico de esta herramienta que permite verificar si determinados puertos TCP admiten conexiones. El comando se instala con `apt install netcat-traditional` pero por simplicidad ya se encuentra preinstalado las máquinas virtuales.

En la siguiente tarea se realizarán conexiones entre 2 máquinas de la misma red. Para realizar las pruebas desde la máquina local es necesario que esté ejecutando Linux o instale WSL de Windows. Otra posibilidad es usar una máquina Linux compartida disponible durante el curso cuyos datos de acceso, están en la página de recursos de la asignatura.

Tarea 14.- Escoja una de las tres opciones para realizar esta tarea:

1. Si está en el laboratorio puede usar el equipo Linux local.
2. Si está usando su equipo con MsWindows instale el soporte WSL2 siguiendo las instrucciones mostradas en el panel de Web de la asignatura. Active OpenVPN e inicie Debian en su equipo.
3. Puede usar la máquina compartida con su *uvus* siguiendo la instrucciones de la T11.3 o T11.4 del Laboratorio 1.

T14.1.- Use un terminal en la máquina y si ha elegido (1) o (2) instale en el equipo *netcat* mediante `sudo apt install netcat-traditional`, para el tercer caso ya está instalado. Si está utilizando Windows con WSL instale también lo siguiente: `sudo apt install openssh-client`

T14.2.- Use el terminal en el equipo para poner *netcat* a escuchar el puerto TCP 9000. El comando es `nc -l -p 9000`, tras su ejecución el terminal parecerá que está bloqueado, pero no es así, está esperando datos entrantes en el puerto 9000 para mostrarlos por la salida.

T14.3.- Para listar los programas que están escuchando en diferentes puertos utilice otro terminal y ejecute el comando `ss -lptn`. Localice en la salida del comando anterior el programa *nc* escuchando en el puerto 9000 .

T14.4.- Ejecute el comando `ss -ltnp` el cual muestra un listado de puertos TCP donde hay algún programa escuchando. Deberá obtener una salida parecida a la mostrada, pero si está usando WSL y obtiene un error, debe actualizar a WSL versión 2.

```
State    Recv-Q   Send-Q     Local Address:Port      Peer Address:Port   Process
LISTEN   0        1                0.0.0.0:9000           0.0.0.0:*       users:(("nc",pid=565,fd=3))
LISTEN   0        128              0.0.0.0:22             0.0.0.0:*       users:(("sshd",pid=447,fd=3))
LISTEN   0        128                 [::]:22                [::]:*       users:(("sshd",pid=447,fd=4))
```

Código 10. Salida del comando ss.

El comando *ss* será de gran utilidad a lo largo del curso. Es una herramienta para mostrar las conexiones de red del equipo. En el ejemplo anterior se ha utilizado con las opciones -ltnp, accediendo a la ayuda de este comando (`man ss`) verá que tiene multitud de opciones. En el ejemplo mostrado las opciones tienen el siguiente significado: `l` hace referencia *listen*; `t` a *tcp*; `n` a *numeric* y `p` mostrar el proceso. En la respuesta mostrada en el código 10 se ha resaltado la columna local, la cual muestra la IP y el puerto en la que se aceptan conexiones, de forma que, en la última fila se ha resaltado en verde el proceso encargado de procesar las peticiones. La interpretación de cada fila es la siguiente:

- Fila 1: Cuando se reciba una conexión entrante TCP hacia cualquier dirección IP del equipo con destino el puerto 9000 se aceptará la conexión entrante y será manejada por el programa *nc*. Este es el programa que se acaba de usar y permite conexiones entrantes por la consola.
- Fila 2: Cuando se reciba una conexión entrante TCP hacia cualquier dirección IPv4 del servidor (0.0.0.0) con destino el puerto 22, se aceptará la conexión y será manejada por el programa *sshd*.
- Fila 2: Cuando se reciba una conexión entrante TCP hacia cualquier dirección IPv6 del servidor (::) con destino el puerto 22, se aceptará la conexión y será manejada por el programa *sshd*.

T14.5.- Ahora desde ese otro terminal, también en la máquina local, vuelva a usar *netcat* para conectar a un puerto de una máquina remota, en este caso al 9000 mediante el comando `nc localhost 9000`. Tras conectar escriba un texto en un terminal, pulse entrar y compruebe que se transfiere al otro terminal. Pare los procesos usando CTRL+C en alguno de los terminales.

T14.6.- Para finalizar el ejercicio sobre *netcat* repita la prueba conectando dos equipos, el local y su máquina GW. Conecte por SSH a su máquina GW y en un terminal ejecute *netcat* escuchando en algún puerto. En otro terminal desde el equipo local conecte a su GW con *netcat*, será parecido a `nc 192.168.20.X 9000`, y realice la prueba tecleando algún texto y pulsando INTRO para que se envíe la trama.

T14.7.- Repita el ejercicio con algún la IP de algún compañero y observe como los terminales funcionan como un chat.

Tras la prueba con *netcat*, en la siguiente parte del laboratorio debe guardar las reglas creadas en cada tarea en un fichero llamado *no-guiado.nft* situado en el directorio */root/firewall*. Este fichero será un fichero ejecutable para poder cargarlo y comprobar los resultados. Es recomendable añadir comentarios usando el carácter # a las reglas que escriba de cara a la evaluación de este laboratorio.

Tarea 15.- Realice los siguientes ejercicios en un nuevo fichero: /root/firewall/no-guiado.nft. Debe incluir un comentario antes de cada regla referenciando la tarea a la que corresponde de cara a la evaluación y su posterior comprensión. Además, para cada ejercicio debe buscar un modo de comprobar si cada una de las reglas añadidas cumple su cometido.

T15.1.- En el nuevo fichero incluya de nuevo las reglas de *nat-permisivo.nft* y cree una tabla de la familia *inet* con 3 cadenas de tipo *filter* enlazadas los ganchos: *input*, *output* y *forward*. En los siguientes ejercicios deberá poner cada regla en la cadena correspondiente según se trate de paquetes entrantes, salientes o reenviados.

T15.2.- Cree una regla que descarte los mensajes ICMP entrantes hacia GW desde la red interna, debe especificar la interfaz adecuada en la regla (*eth0* o *eth1)* usando como referencia la tabla 9 (pág. 18). Compruebe esta regla usando el comando *ping* desde una de las máquinas internas.

T15.3.- Cree una regla que descarte todo el tráfico ICMP entrante hacia GW. Compruebe de nuevo la regla como crea oportuno.

T15.4.- Sin borrar la regla anterior, añada una regla, en el lugar adecuado, que permita recibir paquetes ICMP a través de la interfaz externa y sólo desde la IP 192.168.20.3. Realice el test correspondiente desde la máquina compartida y desde el equipo local.

T15.5.- Descarte todas las conexiones salientes desde VM1, VM2, VM3 y GW, de correo electrónico SMTP saliente (puerto TCP 25) y añada un contador a estas reglas (necesitará varias reglas). Realice la prueba correspondiente con *netcat* desde VM3 y desde GW comprobando los datos del contador en GW con `nft list ruleset`. Puede usar como destino el servidor de correo del DTE: *teclix.dte.us.es*.

T15.6.- Haga seguimiento de las sesiones entrantes *telnet* desde la interfaz externa y consiga que aparezca un mensaje en el registro del sistema con el texto "*Posible escaneo de puertos*". Para testear esta regla no es necesario tener ningún programa trabajando en el puerto 23 (TELNET). Use como referencia el código 7 (pág. 22).

T15.7.- Realice intentos de conexión al puerto 23 con *netcat* o con el programa *telnet* y analice las entradas de registro en el fichero /var/log/syslog buscando la dirección IP y la dirección MAC de origen del paquete entrante en el puerto 23. Use el comando `less /var/log/syslog` para ver el fichero y pruebe la combinación de teclas SHIFT+F para ver el fichero en tiempo real.

T15.8.- Deniegue las conexiones al puerto 22 de GW sólo desde la red privada. Corresponde a la interfaz conectada las máquinas VM1, VM2 y VM3, por lo que debe realizar la prueba desde una de sus maquinas virtuales internas usando SSH, pero asegúrese que desde su equipo local puede conectar por SSH hacia su IP pública.

T15.9.- Descarte todo el tráfico entrante UDP hacia GW, excepto las conexiones desde el puerto 53 desde los equipos de la red 150.214.0.0/16. Puede necesitar varias reglas y para comprobarlas revise: (1) que sigue operativa la resolución de nombres en GW (comando *host*) y (2) que desde la máquina compartida no puede enviar datos al puerto 53. Para esto último use netcat en modo UDP, escuchando en GW y conectando desde la máquina compartida a ese puerto de su GW..

T15.10.- Descarte todo el tráfico WEB desde la red privada dirigido hacia *The Hidden Wiki* (https://thehiddenwiki.org/). Navegue esa página desde VM3 antes del bloqueo comprobar lo que está

bloqueando y después verificar el bloqueo.

T15.11.- Descarte el tráfico saliente desde la red privada en el rango de puertos 6881-6889, usados habitualmente por el protocolo P2P *BitTorrent*. Busque en Internet que tipo de protocolo es *BitTorrent* y como se suele filtrar ya que al ser P2P necesitará múltiples reglas.

T15.12.- Deniegue el tráfico saliente desde el puerto 25 de GW (correo electrónico SMTP) desde la interfaz externa a los paquetes con las banderas SYN y ACK a la vez. ¿Tiene sentido esta regla? ¿No entra en conflicto con la creada en T15.5.-?

4.6. Caso práctico

Este laboratorio finaliza realizando una configuración del firewall que será usada a lo largo de todo el curso. El objetivo es configurar correctamente la red virtual de provista a cada alumno en la asignatura (figura 8) siguiendo los requisitos indicados en las siguientes tareas. Se debe configurar la red estableciendo con una configuración del firewall de filtrado de paquetes ubicado en la máquina GW y esta configuración será persistente, es decir, aunque la máquina se reinicie las reglas establecidas se cargarán automáticamente.

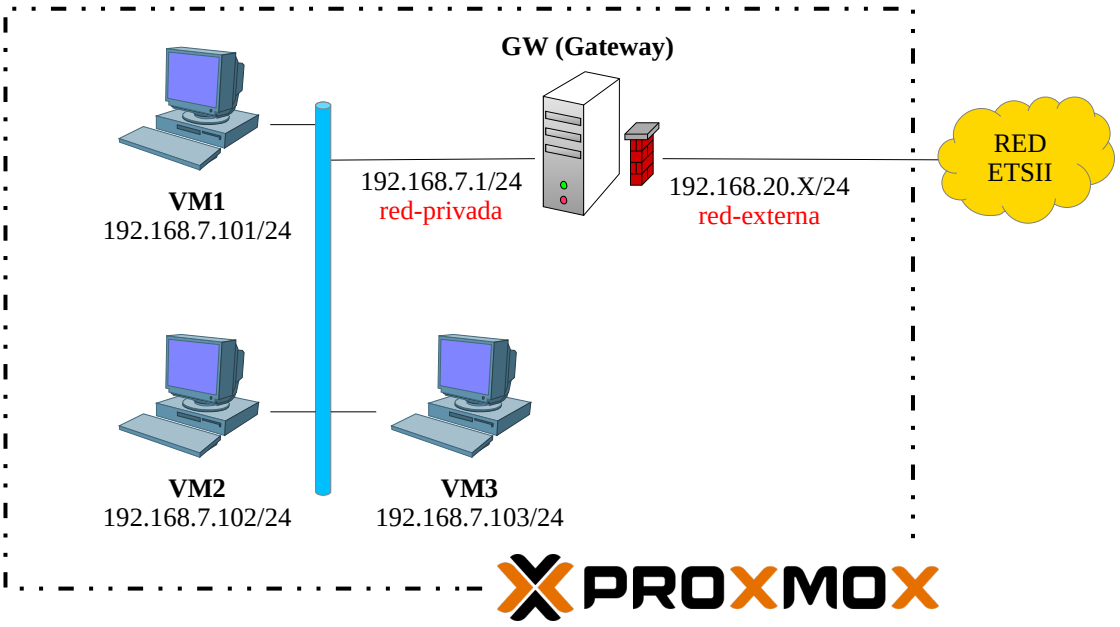


Figura 8. Esquema de configuración de la red virtual.

A partir de aquí, el objetivo es conseguir que las reglas del firewall establecidas sean persistentes tras reiniciar el sistema. Si reinicia la máquina GW observará que las reglas *nftables* no son persistentes, es decir, para cargarlas se debe ejecutar como usuario *root* del script *firewall.nft* (o el que corresponda). Para solventar esto, las distribuciones Linux tienen varios programas para hacer persistentes las reglas de Netfilter. En el caso de servidores basados en Debian, la instalación de nftables añade un servicio al sistema llamado *nftables*, el cual, carga un determinado fichero cuando se inicia el sistema. Está configuración está deshabilitada, en la siguiente tarea se procede a habilitarlo y a preparar el sistema para que cargue en el inicio del sistema el firewall. Respecto a los servicios del sistema, actualmente en Debian estos servicios usan el gestor *systemd*, este gestor se estudia en mayor profundidad en el próximo laboratorio.

Tarea 16.- La distribución Debian siempre incluye una mínima configuración con algunos ejemplos para cualquier software instalado en el sistema. Así, con *nftables* existe un fichero con algunas cadenas creadas pero sin reglas que debe ser editado para personalizar el firewall. Edite con *nano* el fichero /etc/nftables.conf. ¿Cuantas cadenas están creadas y a que ganchos están enlazadas?

T16.1.- Sin borrar el contenido de este fichero, añada al final del mismo una tabla llamada *compartir_internet* con la regla del fichero /root/firewall/nat-permisivo.nft. Copie y pegue entre dos terminales para no volver a escribir las reglas. Cuando termine guarde el fichero y cargue la configuración ejecutando directamente el fichero mediante /etc/nftables.conf. Liste las reglas para asegurarse que se han cargado.

T16.2.- Este fichero se carga automáticamente en el inicio del sistema, si y sólo si, el servicio llamado *nftables* de *systemd* está activado. Aunque *systemd* se trata en el siguiente laboratorio, ahora debe activar este servicio mediante el comando `systemctl enable nftables`, ejecute este último comando como administrador.

T16.3.- Reinicie la máquina GW ejecutando el comando `reboot`, compruebe si tras el reinicio las máquinas internas siguen teniendo conexión a Internet y se han cargado las reglas de netfilter. Para ello, liste las reglas ejecutando en máquina GW el comando `nft list ruleset`.

A partir de aquí se va a establecer una configuración completa del firewall añadiendo las reglas necesarias al fichero */etc/nftables.conf*. Concretamente se trata de determinar qué paquetes estarán permitidos reenviar a través del la máquina GW. En esta primera parte el ejercicio es de filtrado, por ello debe usar la una cadena del tipo *filter* y los paquetes a analizar son los dirigidos a Internet desde las máquinas internas de la red, por ello debe usar la cadena enlazada al gancho *forward* (ver figura 4 pág 8). Todos los paquetes dirigidos al exterior desde la máquinas internas pasarán por esta cadena, por ello, debe añadir en esta cadena las reglas necesarias para que se cumplan los requisitos del ejercicio. Otro aspecto importante de este ejercicio es que se desea crear un firewall prohibitivo para su red interna (todos los anteriores eran permisivos). El procedimiento para conseguirlo es establecer una política predeterminada de descarte (*drop*) para la cadena *forward*. Siguiendo estas indicaciones realice la siguiente tarea.

Tarea 17.- A lo largo de este ejercicio debe probar cada cambio que haga en el fichero verificando si funcionan correctamente los cambios realizados, debe pensar cómo y con que comandos realizar cada testeo.

T17.1.- Edite el fichero /etc/nftables.conf y establezca la política predeterminada a *drop* en la cadena enlazada a *forward*. Recargue la configuración del fichero y compruebe que las máquinas internas VM1, VM2 y VM3 han perdido la conexión a Internet. Asegúrese que la máquina GW no ha perdido la conexión a Internet. ¿Por qué la máquina GW no ha perdido la conexión a Internet y las internas sí?. Use la figuras 2, 3 y 4 (pág. 7) para comprenderlo.

T17.2.- Con la configuración anterior todos los paquetes provenientes de red privada que intentan ser reenviados son descartados por la máquina GW. En primer lugar se desea conseguir que las máquinas de la red privada puedan usar el comando *ping* hacia la red exterior. Cree las reglas necesarias para conseguirlo, considere que serán reglas cuyo veredicto (acción final) sea *accept*. Recargue el fichero y verifique que funciona el comando *ping* desde las máquinas internas (VM1,VM2 y VM3).

T17.3.- Cree dos reglas que permitan el tráfico DNS saliente y entrante desde la red privada. Recuerde que el tráfico DNS utiliza el protocolo UDP en un determinado puerto. Para probar si opera

correctamente desde las máquinas internas use el comando ping o host con un nombre de dominio existente, por ejemplo `ping google.es`, o `host www.dte.us.es` verificando que el sistema resuelve a la IP correctamente.

T17.4.- Se desea refinar más la seguridad en las peticiones DNS ya que son una fuente recurrente de ataques informáticos. Debe crear una definición con un conjunto de direcciones IP correspondientes a servidores DNS, puede usar como ejemplo del código 8 (pág. 23), pero sólo deben aparecer los DNS de la Universidad, los encontrará en panel Web de la asignatura. Cambie las reglas anteriores para que la red privada, sólo se pueda usar los DNS de la Universidad indicados. Verifique el funcionamiento intentando usar un DNS diferente en las máquinas internas, como podría ser 1.1.1.1 u 8.8.8.8.

T17.5.- Para la descarga de software con APT en las máquinas virtuales se está usando un *proxy* para acelerar las descargas de software. Intente actualizar la lista de paquetes `apt update` en la máquina GW y repita el proceso en alguna máquina interna. En el último caso deberá mostrar errores. ¿Ve la dirección IP y el puerto a la que intenta conectar pero no responde?. Debe añadir las reglas correspondientes para permitir el tráfico desde la red privada hacia la IP y puerto del *proxy*.

T17.6.- El siguiente servicio que se desea permitir es la navegación Web. Cree varias reglas para que los equipos de la red privada puedan navegar correctamente por la Web (HTTP y HTTPS).

T17.7.- En las máquinas VM1 y VM2 no está instalado ningún navegador Web y además no existe un escritorio gráfico. Para evitar instalar un escritorio gráfico en esas máquinas, instale el paquete *elinks*, el cual, es un navegador Web en modo texto. Úselo para comprobar si las reglas de la tarea anterior operan correctamente navegando a alguna página Web en modo texto.

T17.8.- Para finalizar el ejercicio, reinicie la máquina GW, y tras el reinicio vuelva a listar las reglas de Netfilter verificando que todos los cambios realizados hasta ahora son persistentes. Compruebe también que las máquinas internas tienen permisos para usar *ping*, DNS, HTTP y HTTPS.

Para terminar este laboratorio se propone añadir algunas reglas cuyo objetivo sea la monitorización pasiva de la red externa. Toda la red 192.168.20.0/24 está dedicada a los servidores Linux de esta asignatura, por lo tanto no debería haber ningún equipo con MsWindows en la misma. Es más es habitual cuando se escanean equipos comprobar los puertos típicos usados en Windows para determinar el sistema operativo, por tanto, consideraremos que si se recibe algún paquete de un determinado protocolo destinado a ciertos puertos estamos siendo escaneados. En línea con esta última afirmación, se propone el un ejercicio de detección de escaneos. En este ejercicio se usará la sentencia include de *nftables*, y hay que considerar que el acceso al sistema de ficheros durante el inicio de un servicio con *systemd* no es completo, esto hace que los ficheros extra de nftables usados durante el inicio del sistema no se puedan ubicar en /root/firewall.

Tarea 18.- Cree un nuevo directorio cuya ruta sea /etc/nftables y después dentro de este directorio cree el fichero /etc/nftables/monitor.nft. En la primera línea debe contener `#!/usr/sbin/nft -f` y **no debe contener** la sentencia `flush ruleset`, ya que será cargado desde otro fichero.

T18.1.- En este nuevo fichero debe crear una tabla de la familia *inet* llamada *scanner*. Añada dentro de esta tabla dos cadenas de tipo *filter* una enlazada al gancho *input* o otra al gancho *output*.

T18.2.- Usando el ejemplo del código 8 (pág. 23) cree definición para un conjunto de puertos referentes a los protocolos TCP y UDP usados por MsWindows en su protocolo de compartición de ficheros e

impresoras (CIFS). Debe buscar estos puertos en Internet, al menos son 2 para TCP y UDP.

T18.3.- Usando las definiciones anteriores, cree reglas que detecten cualquier paquete entrante a estos puertos, pero sólo para la interfaz externa. Usando el ejemplo del código 7 (pág. 22) como base, cada regla debe tener un contador, además debe dejar un mensaje en la bitácora del sistema con el texto *"Posible escaneo de puertos"*, donde claramente se vea la dirección MAC y e IP del origen del paquete.

T18.4.- Testee las reglas anteriores usando la herramienta *netcat*, usando su cuenta de usuario existente en la máquina virtual compartida cuya IP está disponible en la página de recursos de la asignatura.

T18.5.- Como último ejercicio, se desea detectar cualquier intento de entrega de correo electrónico (SMTP) desde GW hacia un servidor externo que no sea realizada por el usuario *root*. Del mismo modo añada una regla con un contador y que deje algún texto en la bitácora del sistema informando del suceso.

T18.6.- Este sistema se desea que esté integrado en el firewall del sistema. Debe editar el archivo `/etc/nftables.conf` y usar adecuadamente la sentencia `include` para que se cargue el fichero desde `/etc/nftables/monitor.nft`. En el ejemplo del código 8 (pág. 23) también se muestra el uso de esta directiva.

T18.7.- Reinicie la máquina GW para verificar que todos los cambios son persistentes y que el sistema de monitorización se ha cargado correctamente.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 3 - Servicios de red

Enunciados de laboratorios
Tecnologías Avanzadas de la Información
Rev. 9 - 10/11/23

1. Introducción y objetivos

La duración estimada de esta sesión de laboratorio es de **8 horas**. El propósito general de esta sesión de laboratorio es realizar un despliegue de diferentes servicios entre los equipos de la red virtual. Este despliegue será necesario para realizar el último laboratorio dedicado al *Control de Tráfico* y *Calidad de Servicio*.

Concretamente se instalará un servidor Web, un servidor SMB, un servidor FTP y un servidor DNS. Los servicios instalados serán públicos a través de la máquina que hace de puerta de enlace (GW). Para conseguirlo se propone configurar adecuadamente *netfilter* de Linux y realizar el reenvío de tráfico a la máquina que sirve cada uno de los servicios.

Nombre de fichero	Descripción
vm1.crt, vm1.pem	Certificado digital con clave pública, clave privada
ev.zip	Clonación simple de la página de enseñanza virtual
ev.us.es.crt, ev.us.es.pem, tai-ca.crt	Certificado digital con clave pública, clave privada y Autoridad de Certificación

Tabla 1. Ficheros necesarios para la realización del laboratorio.

2. Servicios con Systemd

Antes de abordar este laboratorio es conveniente estudiar como funciona el inicio del sistema en las distribuciones Linux actuales. En este sentido, actualmente la mayoría de las distribuciones administran los

servicios del sistema mediante un software de control llamado *systemd* [systemd.io]. Aunque este sistema no está ausente de duras críticas por parte de los desarrolladores, actualmente ha sido adoptada con reticencias por distribuciones como Debian (ver discusión en [VotaciónSystemd]). A lo largo del curso se hará un uso continuo de este administrador de servicios, por lo que se presenta su uso básico a continuación.

Tarea 1.- El comando básico para administrar sistemas Linux con *systemd* es `systemctl`. Ejecute este comando como administrador en GW y verá una lista de servicios donde puede desplazarse con los cursores. Busque el servicio *ssh*, para ello pulse / (mayúsculas+7) y aparecerá en la parte baja el cursor donde puede teclear *ssh*.

T1.1.- Para salir del listado de servicios pulse `q` y volverá a la línea de comandos. Ejecute el comando anterior pero con el parámetro indicado: `systemctl status`. Verá los servicios en ejecución en forma de árbol, indicando que servicio inicio otro servicio. De nuevo debe salir con la tecla `q`.

T1.2.- Acceda al escritorio de VM3 y repita el comando `systemctl status`. Compare la salida con GW, ¿por qué VM3 tiene servicios en ejecución?.

T1.3.- Otra opción fundamental en el control de *systemd* es poder consultar el estado de un servicio concreto. Trabajando de nuevo en GW pruebe a ejecutar `systemctl status ssh` e interprete la salida.

T1.4.- El inicio de todos los servicios de un sistema operativo es una tarea compleja, donde entran en juego dependencias entre los propios servicios. Por ejemplo, para iniciar *nftables* es necesario que la red esté activa, por ello, debe iniciarse posteriormente a la red. Para ver el árbol de dependencias completo use el siguiente comando: `systemctl list-dependencies`.

T1.5.- Ahora vea el árbol de dependencias de *nftables* por separado con `systemctl list-dependencies nftables`.

T1.6.- Dada la complejidad de este sistema, se acompañan herramientas como *systemd-analyze* capaz de generar gráficos de inicio del sistema y dependencias. Trabajando en VM3, ejecute `systemd-analyze critical-chain` y de manera opcional vea la ayuda de este comando con `systemd-analyze -h` o `man systemd-analyze`. Intente generar una imagen en formato *.svg* con el gráfico temporal de inicio del sistema, para verlo abra el fichero con Firefox en VM3.

T1.7.- Otra característica controvertida de *systemd* es la forma en la que se guarda el registro de sucesos del sistema. En Linux la filosofía es mantener en la mayor medida de lo posible los ficheros legibles en modo texto, como es por ejemplo el fichero */var/log/syslog* y todos los de ese directorio. En cambio, *systemd* los guarda en binario y es necesario usar el comando journalctl para verlo. Ejecute este último comando y avance por la información mostrada.

T1.8.- Habrá observado que contiene multitud de información de eventos de todos los servicios del sistema. Para poder filtrar por un servicio se usa la opción `-u`, úsela probando este comando: `journalctl -u ssh`

T1.9.- Por último para ver el registro en tiempo real deje en un terminal ejecutando el comando `journalctl -f` y entre por SSH de forma remota para observar los cambios.

3. Instalación de servicios

Para realizar correctamente este laboratorio se recomienda instalar el software de la tabla 2 en el equipo indicado. Recuerde que la instalación se realiza mediante el gestor de paquetes APT.

Nombre del paquete	Descripción	Ubicación
firefox	Navegador Web	Máquina VM3
filezilla	Programa con interfaz gráfica para transferencias FTP/FTPS/SFTP	Máquina local
ftp	Programa en línea de comandos para trasferencias FTP (opcional)	Máquina VM3 y local

Tabla 2. Paquetes recomendados para la realización del laboratorio.

Los servicios que se instalarán en el entorno virtual son: servidor Web y FTP en la máquina VM1; servidor SMB y DNS en la máquina VM2. Para estos servicios, la distribución de Linux utilizada ofrece diferentes alternativas, en la tabla 3 se presentan las soluciones más habituales, pero debe usar la que se indique en este laboratorio en la tarea correspondiente.

Tipo de Servidor	Nombre del paquete	Página oficial
SMB	samba	https://www.samba.org/
Web	apache2 (★★★)	https://httpd.apache.org/
	lighttpd	https://www.lighttpd.net/
	nginx	https://nginx.org/
FTP	proftpd-basic (★★★)	http://www.proftpd.org/
	vsftpd	https://security.appspot.com/vsftpd.html
	pure-ftpd	https://www.pureftpd.org/
DNS	bind9 (★★★)	https://bind.isc.org/
	unbound	https://unbound.docs.nlnetlabs.nl/en/latest/

Tabla 3. Servidores de uso común en Linux.

En este laboratorio se usarán los indicados con 3 estrellas, no los instale ahora, espere a cada tarea para realizar la instalación de manera adecuada siguiendo las instrucciones.

3.1. Instalación de un servidor Web con soporte SSL

Respecto al servidor Web se deben ofrecer 2 servicios: uno en el puerto estándar HTTP (puerto 80) para páginas sin conexión segura, y otro para HTTPS (puerto 443) donde se servirán páginas seguras usando SSL/TLS. Para la puesta en marcha de este servicio realice los siguientes pasos:

Tarea 2.- Instale el servidor Web *Apache2* en la máquina VM1, debe realizar la instalación con APT. Compruebe tras la instalación que hay un programa escuchando en el puerto 80, use el comando `ss -ltnp` e interprete correctamente la salida.

T2.1.- Desde la máquina VM3 navegue con Firefox y verifique si aparece una página Web en la dirección http://192.168.7.101.

T2.2.- La página Web que se está sirviendo está en la ubicación /var/www/html. Cambie dicha página de forma que cuando se navegue aparezca un mensaje de bienvenida con su nombre, del tipo *"Bienvenido al servidor web de Ana y mi uvus es"*. Es obligatorio que aparezca el *uvus* en la página Web para la evaluación de esta tarea.

Tarea 3.- El siguiente paso es activar SSL en el puerto 443 utilizando una configuración básica predeterminada. Para activar el soporte SSL con S*pache2* en primer lugar active el módulo SSL mediante la ejecución del comando `a2enmod ssl`.

T3.1.- Tras esto, en el directorio /etc/apache2/sites-available existe un fichero con un ejemplo de configuración SSL, debe enlazarlo o copiarlo al directorio /etc/apache2/sites-enabled.

T3.2.- Copie los ficheros vm1.crt y vm1.pem suministrados con el material del laboratorio a /etc/apache2 y edite el fichero de configuración SSL de Apache2 de la tarea anterior. Establezca correctamente el certificado digital y la clave privada interpretando las directivas de este fichero. Ayúdese viendo con un editor de textos los ficheros vm1.crt y vm1.pem, averiguará cual es el contenido de cada uno de ellos.

T3.3.- Tras guardar el archivo de configuración que ha editado, use el comando `apache2ctl configtest` para comprobar si ha cometido errores en la configuración. Debe mostrar sólo una advertencia referente a la directiva *ServerName*.

T3.4.- Aunque no es necesario, para solucionar este último aviso puede editar el fichero /etc/apache2/apache2.conf añadiendo al final del mismo como *ServerName* el nombre de la máquina, es decir, añada al final de este fichero una nueva línea conteniendo: `ServerName vm1`

T3.5.- Tras ello reinicie Apache con el comando `systemctl restart apache2` y verifique de nuevo con el comando `ss -ltnp` que el servidor Web Apache2 está escuchando en el puerto TCP 443. En caso de fallo debe leer los errores en los ficheros del directorio /var/log/apache2.

T3.6.- Navegue desde la máquina VM3 a la dirección http://192.168.7.101/ y compruebe que sigue operativa la navegación no segura HTTP. Tras esto navegue usando el protocolo HTTPS para comprobar que ocurre con los certificados digitales. ¿Sabe interpretar el aviso del navegador?

T3.7.- Antes de confirmar la excepción de seguridad en Firefox use el botón View para revisar los datos del certificado y ver donde está el problema.

3.2. Instalación de un servidor de ficheros para SMB

Server Message Block (SMB) es un protocolo creado por IBM para acceder por red a archivos y directorios. Este protocolo está incluido en MsWindows como sistema predeterminado para compartir carpetas y archivos en la red. La implementación de SMB realizada por Microsoft se llama *Common Internet*

File System (CIFS) y difiere en cierta medida de la versión inicial. Linux soporta este protocolo mediante el software de código abierto llamado *Samba*.

El paquete Samba permite diversos tipos de configuración para el acceso a las carpetas compartidas. En este laboratorio se usaran dos modos fáciles de configurar. El primero usará los usuarios locales de la máquina para crear carpetas personales privadas para albergar ficheros. En el segundo ejemplo se creará una carpeta pública para que se pueda acceder sin ningún tipo de credenciales.

Tarea 4.- En la máquina VM2 instale el paquete *samba* mediante el gestor de paquetes APT (comando: `apt install samba`).

T4.1.- Revise los puertos en los que *samba* está escuchando, tanto TCP como UDP. Debe usar los comandos `ss -ltnp` y `ss -lunp`. Serán necesario posteriormente para configurar correctamente el firewall.

T4.2.- Edite la configuración de Samba en /etc/samba/smb.conf y busque la sección donde se configuran las cuentas de los usuarios del sistema correspondiente a la sección [homes]. Debe activar la visibilidad cambiando `browseable = yes` y también activar el soporte de escritura `read only = no`. Cuando termine reinicie el servicio Samba mediante `systemctl restart nmbd.`

T4.3.- Desde la máquina VM3 utilice el administrador de ficheros de escritorio para conectar mediante el protocolo SMB. Para ello debe escribir en la barra de ubicación smb://192.168.7.102 obteniendo lo mostrado en la figura 1.

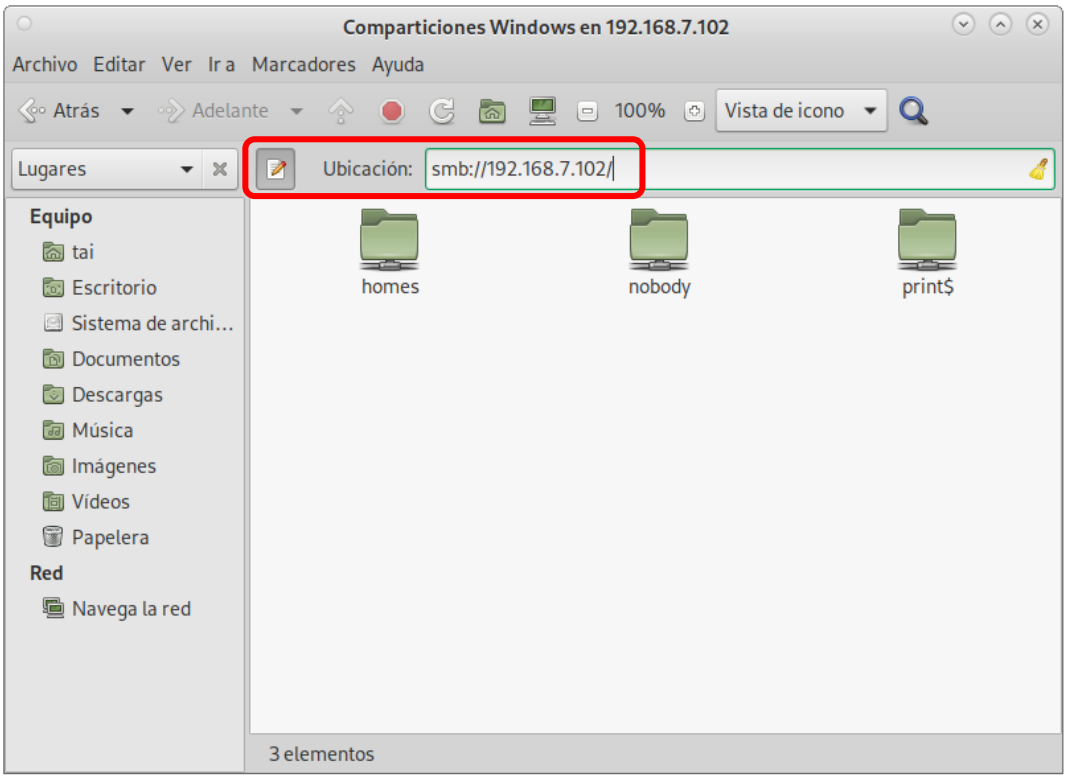


Figura 1. Conexión remota SMB mediante el administrador de ficheros.

T4.4.- Si intenta entrar en la carpeta homes le solicitará credenciales pero, aunque intente entrar con el usuario *tai* y su clave, el sistema no le dejará. De manera predeterminada ningún usuario local de VM2 tiene acceso mediante a ninguna carpeta. Hay que autorizar explícitamente cada usuario local en Samba.

T4.5.- Como administrador en VM2 use el comando `smbpasswd -a tai` para dar de alta el usuario en *Samba* con una clave. Tras finalizar repita el proceso de T4.3.- para ver si puede acceder a la carpeta e intente copiar archivos arrastrándolos para asegurarse que funciona correctamente.

T4.6.- Desde la línea de comandos en VM2 entre en el directorio /home/tai y asegúrese que aparecen los ficheros copiados en la tarea anterior.

Se ha comprobado que Samba de forma predeterminada no permite a ningún usuario existente en el sistema usar los servicios SMB. Es necesario que el administrador del sistema añada a los usuarios explícitamente y con una clave, la cual, no tiene que ser la misma que la del sistema. En el siguiente ejemplo se pretende añadir usuarios sólo al servicio SMB, es decir, no podrán entrar mediante SSH en la máquina y ejecutar comandos.

Tarea 5.- Añada un nuevo usuario al sistema operativo llamado *taiwindows* VM2, el comando es: `adduser taiwindows.` Compruebe que el nuevo usuario puede acceder de forma remota desde la máquina GW mediante SSH con su clave, comando `ssh taiwindows@192.168.7.102`.

T5.1.- Dé de alta al usuario *taiwindows* en Samba repitiendo el procedimiento seguido en T4.5.- y verifique que puede acceder a sus ficheros repitiendo el procedimiento usado en T4.3.-

T5.2.- Para proceder al denegar el acceso mediante SSH se recomienda quitar el *shell* (intérprete de comandos) al usuario. La mejor forma de hacerlo es mediante el comando usermod. Busque en Internet como quitar el shell a *taiwindows* mediante este comando.

T5.3.- Tras el cambio anterior compruebe que el nuevo usuario no puede entrar mediante SSH al sistema desde la máquina VM3, pero que desde el administrador de ficheros puede seguir copiando ficheros mediante SMB.

T5.4.- Cree una carpeta en /srv/public y compártala mediante el nombre *public*. Debe editar el fichero /etc/samba/smb.conf creando al final una sección que comience por [public] con las opciones adecuadas para que tenga el siguiente comportamiento:

- Cualquier usuario sin necesidad de identificarse puede escribir y leer archivos en la carpeta.
- Sea visible cuando se accede a smb://192.168.7.102/

T5.5.- Opcional-a: Cree una carpeta llamada en /srv/archivo que se comparta con el nombre *archivo* de forma que:

- Cualquier usuario sin necesidad de identificarse pueda leer archivos de la carpeta.
- Sólo el usuario *taiwindows* puede escribir en la carpeta.
- Sea visible cuando se accede a smb://192.168.7.102/.

T5.6.- Opcional-b: Cree otra carpeta en /srv/interna que se comparta con el nombre *interna* de forma que:

- Solo los usuarios con credenciales puedan leer la carpeta.

- Sólo el usuario *tai* puede escribir en ella.
- Sea visible cuando se accede a smb://192.168.7.102/.

3.3. Instalación de un servidor FTP

Para el servidor FTP se usará *proftpd-basic*. Tras la instalación de este servidor FTP podrá comprobar si el servidor opera, realizando conexiones con las credenciales de los usuarios ya existentes en el sistema, considere que esta configuración predeterminada es un potencial agujero de seguridad por diversos motivos:

- La conexión FTP no está cifrada, además el usuario es el mismo que el del sistema, por tanto, si el inicio de sesión es capturado, con esa misma identificación se puede entrar mediante SSH y obtener un *shell* en la máquina.
- Por defecto se sirve como directorio la cuenta del usuario (ruta /home/nombre_usuario), pero es posible acceder a todo el árbol del sistema de ficheros, es decir, la conexión FTP no está limitada a la cuenta del usuario que ha accedido.

Independientemente de estos problemas de seguridad se propone la instalación del servidor FTP y el cambio de configuración predeterminada para intentar evitar, en la medida de lo posible, los problemas de seguridad descritos.

Tarea 6.- Instale el servidor FTP *proftpd-basic* mediante APT en la máquina VM1.

T6.1.- Tras la instalación, compruebe mediante el comando `ss -ltnp` el listado de programas que están escuchando en los puertos TCP del sistema. Debe aparecer en el puerto 21 escuchando el servidor FTP.

T6.2.- Instale el cliente *ftp* en la máquina VM1 y VM3 (`apt install ftp`) y úselo desde VM1 para conectarse a sí mismo del modo `ftp localhost` indentificándose con el usuario *tai*. Ejecute el comando `help` para ver los comandos disponibles en el protocolo FTP. ¿Es capaz de crear un directorio usando la línea de comandos FTP?. Repita la prueba desde la máquina VM3 usando el programa Filezilla. Debe instalar este programacon APT y transferir archivos de VM3 a VM1.

T6.3.- En VM1 añada un usuario llamado *web*, con clave *tai* (para facilitar la evaluación), mediante el comando `adduser --home /var/www web`. Compruebe si puede conectarse mediante FTP realizando pruebas desde la máquina VM3. Una vez conectado intente crear un directorio o transferir archivos y observará un error.

T6.4.- Desde la máquina VM3 entre por SSH en la máquina VM1 con el usuario *web* mediante `ssh web@192.168.7.101` e intente de nuevo crear un directorio y observe el error. Liste el directorio con el comando `ls -la` y observe quien es el propietario y cuales son los permisos de los archivos y directorios de /var/www.

T6.5.- De nuevo mediante `ssh web@192.168.7.101` compruebe si puede usar el comando `sudo su` para convertirse en administrador del sistema.

T6.6.- Usando *Filezilla* desde su máquina VM3, conecte con el usuario *web* y navegue a la raíz del sistema de ficheros de la máquina remota. Entre en el directorio */etc* y compruebe si tiene acceso a algunos ficheros de configuración del sistema.

En las tareas T6.3.- y T6.4.- no fue posible alterar el contenido de la carpeta *home* del usuario *web* ya que en T6.3.- se estableció a un directorio ya existente /var/www. Cuando listó este último directorio se debió obtener lo siguiente:

```
ls -la /var/www/
total 12
drwxr-xr-x 3  root root 4096 jul 28 11:55 .
drwxr-xr-x 12 root root 4096 jul 28 11:55 ..
drwxr-xr-x 2  root root 4096 jul 28 11:57 html
```

La línea resaltada en rojo corresponde al directorio /var/www y muestra que el propietario es el usuario *root*, el grupo de usuarios con el que se comparte también es el grupo *root* y sólo tiene permiso se escritura (w) el usuario *root*. Los permisos indican que tanto para el grupo como para otros usuarios aparece "-" indicando que no tiene permiso de escritura. Un objetivo de las siguientes tareas será conseguir que el usuario *web* pueda modificar el contenido de /var/www sin restricciones.

Por otro lado, en la última prueba se accedió a los archivos de configuración del sistema operativo. Aunque los permisos predeterminados establecidos por la distribución Debian impiden que los usuarios puedan acceder a muchos de ellos, un error del administrador de la máquina en este sentido podría permitir acceder a alguno crítico violándose la seguridad del sistema. Además el sistema permite la conexión por SSH por lo que el usuario puede ejecutar procesos en la máquina, en un servidor en explotación se debe evitar que los usuarios con acceso FTP puedan conectar mediante SSH y obtener un *shell*. Se pretende aumentar la seguridad del servicio FTP de la siguiente forma:

1. Para evitar que se pueda acceder por SSH y ejecutar comandos se desactivará el servicio SSH al usuario web.
2. Para aumentar la seguridad se propone en primer lugar configurar el servidor FTP para que enjaule (*chroot*) a los usuarios en su directorio personal. Enjaular consiste en que los usuarios no puedan acceder a niveles de directorio anteriores a su directorio de inicio, es decir, no podrá acceder a ninguna carpeta de nivel superior. Internamente consiste en que el sistema considere que la raíz del sistema de ficheros es un determinado directorio, por tanto, sólo puede acceder a los directorio hijos, pero nunca al directorio padre ni superiores.
3. El usuario *tai* si puede entrar por SSH para administrar la máquina pero se desactivará FTP para este usuario, ya que es el administrador.

El procedimiento para realizar todo esto es bastante similar al hecho con Samba como podrá observar en la siguiente tarea.

Tarea 7.- Ejecute el comando `systemctl` sin argumentos para ver todos los servicios del sistema y busque mediante "/" *ftp* para ver el nombre del servicio que *proftpd-basic* ha instalado. Use este nombre para ver el estado o reiniciar el servicio mediante `systemctl status [nombre-servicio]` o `systemctl restart [nombre-servicio]`. Tendrá que ejecutar este comando cada vez que se indique en esta tarea.

T7.1.- Configure el servidor FTP para que enjaule (*chroot*) al usuario en su directorio personal. Debe editar el fichero de configuración que está en /etc/proftpd/. Busque como se realiza la configuración con

la opción *chroot* del servidor *proftpd* y pruebe la configuración reiniciando el servicio con systemctl en cada prueba.

T7.2.- Una vez configurado el *chroot*, se desea que el usuario *web* pueda alterar todas la páginas Web del servidor, para ello hay que establecer los permisos de la carpeta /var/www adecuadamente para que el usuario *web* pueda: leer, escribir y crear directorios. Todo lo anterior no debe afectar a que el usuario www-data pueda leer ficheros y entrar los directorios. Esta configuración tiene múltiples soluciones, debe usar los comandos `chmod`, `chown`, `chgrp` y si lo cree necesario use el permiso s (*sticky*).

T7.3.- El siguiente paso es prohibir conexiones SSH al usuario *web*, para ello siga el mismo procedimiento hecho cuando se configuro Samba (ver T5.2.-). Tras este cambio verifique que se ha anulado tanto el acceso SSH como el FTP. Para solucionarlo, en el archivo de configuración proftpd.conf busque y active una directiva que permita la conexión FTP de usuarios sin shell válido.

T7.4.- Por último se desea deshabilitar el acceso FTP al usuario *tai* pero no el acceso SSH. Edite con el fichero /etc/ftpusers y siga las instrucciones que se indican en la primera línea de ese fichero para conseguirlo.

T7.5.- Opcional-3c: Usando la documentación del servidor FTP que ha instalado intente activar el soporte STARTTLS para conseguir que la comunicación se cifre. Configúrelo usando los mismos ficheros de certificado y clave privada usadas con el servidor Web.

4. Configuración de la puerta de enlace

Tras el despliegue de los servicios en la red interna hay que configurar el firewall de la máquina GW de forma que se hagan públicos ciertos servicios redirigiendo el tráfico a los equipos internos que gestionan cada uno de ellos. En la figura 2 se muestra esquemáticamente cómo debe configurarse el firewall. Esta figura representa la distribución de los servicios en máquinas de la red interna, el objetivo en el despliegue del laboratorio consiste en servir cada servicio mediante una máquina virtual, para ello, es necesario redirigir tráfico desde la máquina GW a los equipos internos. Así, trabajando en la máquina GW se añadirá en *nftables* reglas adecuadas en determinadas cadenas. Para ello, se creará un nuevo fichero /etc/nftables/sevicios.nft que será incluido desde el fichero principal /etc/nftables.conf.

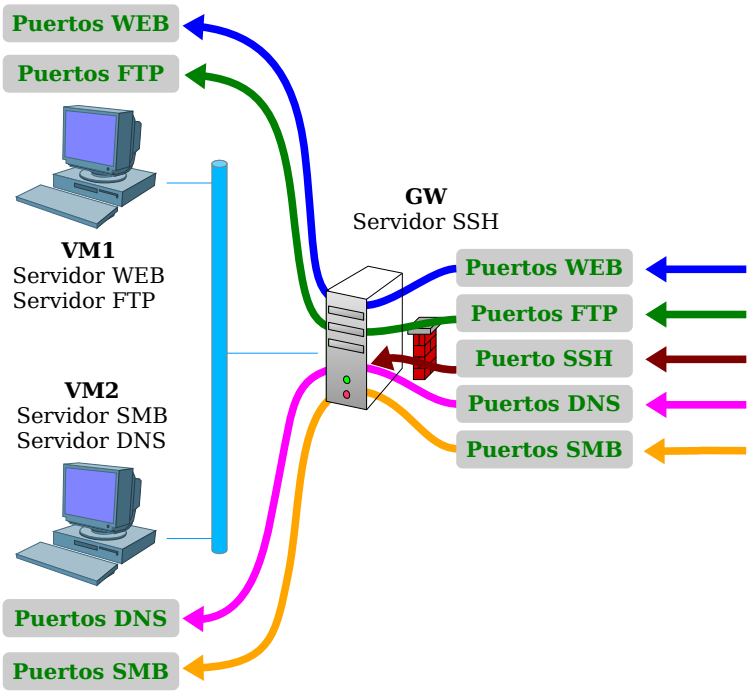


Figura 2. Esquema de configuración del firewall.

Tarea 8.- Como administrador en GW cree un nuevo fichero /etc/nftables/servicios.nft e inclúyalo adecuadamente desde el fichero /etc/nftables.conf.

T8.1.- Trabajando en este nuevo fichero /etc/nftables/servicios.nft cree una nueva tabla llamada servicios con una cadena llamada servicios_dnat.

T8.2.- La cadena debe operar de la forma indicada en la figura 2, redirigiendo puertos entrantes a máquinas internas mediante *Destination NAT* (DNAT). Para ello debe poner en la cadena anterior el tipo nat y el gancho adecuado, ¿a qué gancho de *netfilter* se deben enlazar las cadenas con acciones de tipo *dnat*?

T8.3.- En primer lugar se debe redirigir el tráfico HTTP entrante a GW hacia la máquina VM1. Escriba una regla que, para el tráfico dirigido a su IP pública (192.168.20.X) hacia el puerto 80, sea redirigido al puerto 80 de la máquina VM1.

T8.4.- Navegue a la dirección http://192.168.20.X desde un ordenador la red externa (equipo local, máquina compartida) para verificar que aparece la página Web con su nombre. Repita el proceso desde VM3 para comprobar que también es visible la página.

T8.5.- Repita el proceso con el puerto correspondiente a HTTPS de forma que todo el tráfico entrante HTTPS se redirija también a su máquina VM1.

T8.6.- Repita la navegación a la dirección https://192.168.20.X desde una máquina externa y desde VM3.

Ahora el objetivo es redirigir el tráfico FTP a la máquina VM1, pero esta tarea presenta cierta dificultad puesto que el protocolo FTP utiliza dos conexiones TCP para operar, una de control y otra de datos. El funcionamiento de este protocolo se puede resumir de la siguiente forma:

1. El cliente FTP inicia una conexión TCP hacia el puerto 21 del servidor. Usando esta conexión se le solicita al cliente su identificación con un usuario y clave.
2. Superado el paso anterior, la conexión hacia el puerto 21 permanece abierta y se llama canal de control. En este canal de control se pueden enviar comandos al servidor.
3. En el momento que el cliente envía el primer comando al servidor, la respuesta se recibirá por una nueva conexión TCP llamada conexión de datos. Para iniciar la conexión de datos existen dos posibilidades:

   3.1. Modo activo: Este es el modo predeterminado y en este caso el servidor inicia una nueva conexión desde su puerto origen 20 hacia la IP del cliente usando un puerto aleatorio, previamente negociado.

   3.2. Modo pasivo: (1) El cliente indica que desea usar este modo enviando el comando PASV; (2) el servidor responde por el canal de control indicando un número de puerto donde espera recibir una conexión; (3) el cliente inicia una conexión TCP hacia el puerto indicado por el servidor.
4. Una vez establecido el canal de datos se pueden transferir ficheros mediante este protocolo.

El modo activo es fácil de configurar pero cuando el cliente está detrás de un firewall o NAT, este modo no suele operar correctamente. En cambio, el modo pasivo, aunque más complejo de configurar, es el más usado ya que facilita la conexión de clientes situados detrás de un firewall o NAT. Para solventarlo existen módulos de netfilter dedicados a manejar este tipo de conexiones, aunque no se usarán en este laboratorio y se intentará configurar el servicio acotando los puertos utilizados en el servidor. En primer lugar se configurará el servidor FTP en modo pasivo añadiendo reglas DNAT que redirijan rangos de puertos adecuadamente.

De forma predeterminada los clientes inician el canal de datos en modo activo y en caso de error pasan a pasivo automáticamente enviando el comando PASV por el canal del control (puerto 21). Cuando el servidor recibe el comando PASV, la iteración cliente/servidor en modo pasivo es la descrita en la figura 3, donde se puede observar la secuencia de conexión del canal de datos en color rojo. Como ya se indicó anteriormente, la secuencia mostrada en la figura es la siguiente:

1. El cliente inicia una conexión hacia el servidor en el puerto 21 y, tras superar la identificación de usuario y contraseña, envía el comando PASV al servidor por esta conexión (canal de control).
2. El servidor responde por el mismo canal (puerto 21) indicando al cliente un número de puerto, diferente al 21, donde el servidor está esperando una nueva conexión entrante desde el cliente que, una vez establecida será el canal de datos.
3. El cliente inicia una nueva conexión hacia el servidor hacia el puerto indicado por el servidor en el paso anterior.

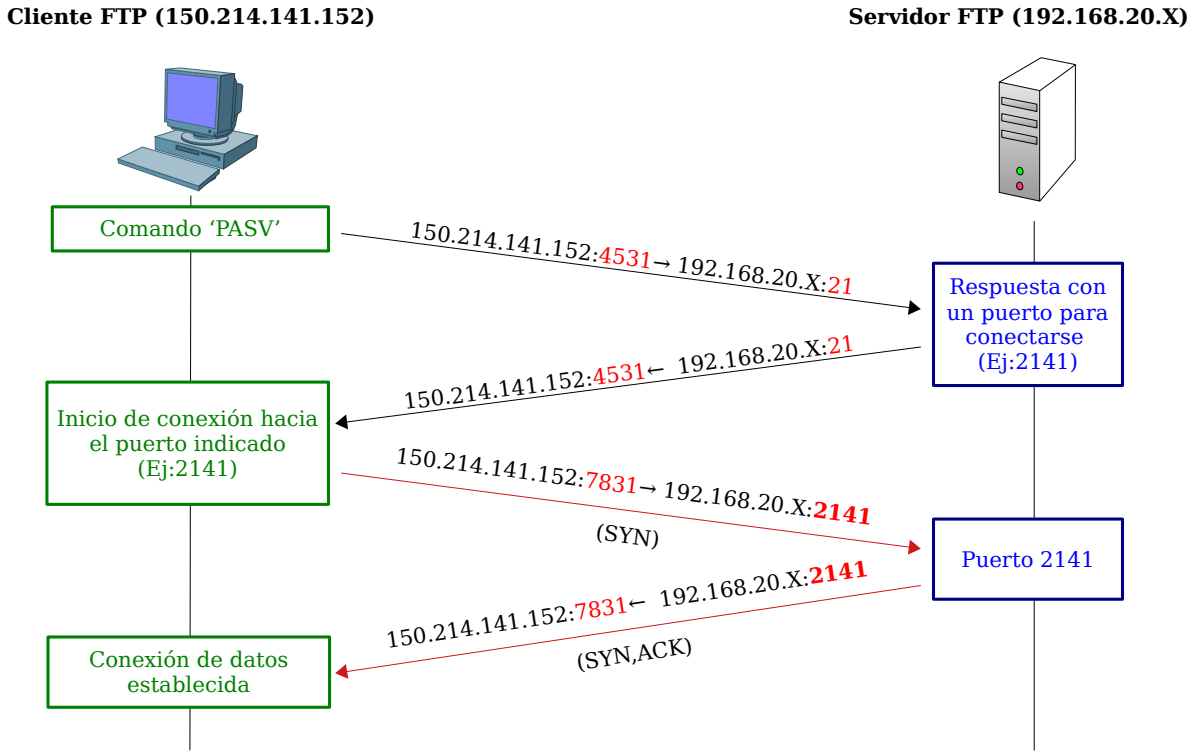


Figura 3. Esquema de conexión FTP en modo pasivo.

Para la secuencia anterior el principal problema reside en que el *firewall* debe saber que puerto indica el servidor al cliente en el paso 2 y aplicar una regla DNAT para redirigir este puerto al servidor FTP. Existen multitud de soluciones para este problema, como son módulos dedicados de *netfilter* y *nftables* para el protocolo FTP (*nf_conntrack_ftp*, *nf_conntrack_helper*) u otras opciones como la redirección de todos los puertos, esta última no muy aconsejable. En este laboratorio se usará una solución simple, pero que depende en gran medida del servidor FTP usado. Concretamente en la solución propuesta se utilizará *proftpd* y se configurará de forma que responda al comando PASV del cliente en un rango de puertos definidos por el administrador del servicio FTP. De esta forma sólo habrá que configurar el *firewall* para que redirija este pequeño rango de puertos.

Tarea 9.- Siguiendo el esquema de la figura 3, se pretende poner operativo el modo pasivo desde la red exterior. En primer lugar añada una regla DNAT similar a las hechas para HTTP que, el tráfico entrante dirigido a su IP pública y puerto 21, sea redirigido hacia el puerto 21 de la máquina VM1.

T9.1.- Para comprobar si funciona intente conectar desde la máquina local o la máquina compartida con el comando `ftp 192.168.20.X` (no use Filezilla) pero verá que no consigue conectar. Añada un contador a esta última regla e intente de nuevo la conexión. Tras el último intento ejecute `nft list ruleset` para ver que el contador de la regla DNAT se incrementa, lo que significa que la regla opera correctamente. Si la regla operaba correctamente, ¿por qué no hay conexión exterior TCP en el puerto 21 con la máquina VM1 a través de la máquina GW?

T9.2.- Para entenderlo, revise el camino que siguen los paquetes DNAT en la figuras de *netfilter* del laboratorio anterior. ¿En qué punto del camino se están filtrando los paquetes entrantes?. Debe añadir reglas en alguna cadena ya existente para permitir tráfico por el puerto 21 y debe estar limitado

únicamente, hacia y desde, la máquina VM1.

T9.3.- Cuando lo consiga, el cliente FTP le pedirá un usuario y debe usar el usuario *web* con la clave correspondiente. Una vez dentro, intente listar los ficheros con el comando `ls` y verá que no funcionará la orden de listado de directorio, aborte la conexión FTP de la consola con CTRL+C. El bloqueo es debido a que el servidor FTP intenta abrir otra conexión TCP para transmitir datos y el firewall de GW lo rechaza.

T9.4.- Se procederá a configurar el modo pasivo del servidor *proftpd* para que responda a la secuencia mostrada en la figura 3 en un rango de puertos. Edite el fichero de configuración /etc/proftpd/proftpd.conf y busque las directivas PassivePorts y MasqueradeAddress, debe establecer en ellas un pequeño rango de puertos, del 2048 al 2303 y la dirección IP externa 192.168.20.X respectivamente. Tras este cambio debe reiniciar el servicio con *systemctl*.

T9.5.- Del mismo modo que ha redirigido en la tarea anterior el puerto 21, deberá también añadir reglas extra para aceptar tráfico en el rango de puertos 2048-2303 y redirigirlos a la máquina VM1. Para trabajar con rangos de puertos debe buscar información sobre *nftables* en la red.

T9.6.- Cuando crea que lo tiene operativo, repita la conexión FTP desde la máquina local o compartida mediante mediante la siguiente secuencia de comandos:

1. Ejecute el comando `ftp 192.168.20.X` y use las credenciales *web* para entrar.
2. Ejecute el comando FTP `passive` y asegurese es estar en ese modo antes de seguir.
3. Ejecute el comando `ls -l` y verifique que puede ver los ficheros y directorios remotos.
4. Pruebe también el programa *Filezilla* conectando desde un equipo externo en modo pasivo a su servidor FTP y compruebe si es capaz transferir ficheros con este programa.

T9.7.- Desde las máquinas internas VM1, VM2 y VM3 realice las siguientes pruebas debería funcionar tanto usando la dirección IP privada de VM1 como la IP pública de GW haciendo las siguientes pruebas:

1. Ejecute el comando `ftp 192.168.7.101` y use las credenciales *web* para entrar.
2. Ejecute el comando `ftp 192.168.20.X` y use las credenciales *web* para entrar.

T9.8.- Opcional-3d - Desde la máquinq GW no funciona `ftp 192.168.20.X`, para solucionarlo hay que crear reglas adicionales. Para más información busque en Internet información sobre *NAT-Reflection* o *NAT-Loopback*.

Como resumen y para comprobar si está correctamente realizada la puesta en marcha del servidor FTP, debería ocurrir lo siguiente, haciendo las comprobaciones adecuadas:

- Desde la red exterior, esto es, ordenador del aula, ordenador personal con la VPN activa o la máquina compartida, operará correctamente FTP en modo pasivo, y se puede comprobar en línea de comandos con el comando *ftp* o con Filezilla.
- Desde la red interna el modo activo operará correctamente siempre que la IP destino sea 192.168.7.101 y desde VM1, VM2 y VM3 debe funcionar el pasivo usando la IP pública 192.168.20.X
- Desde la máquina GW funcionará la IP pública como destino sólo si ha realizado la tarea opcional.

Tarea 10.- Trabajando de nuevo en el firewall, debe redirigir y permitir el **mínimo número de puertos** necesarios para que el servidor Samba de VM2 sea accesible desde la red externa. Para verificar la conexión puede probarlo desde la máquina compartida usando su IP pública mediante smb://192.168.20.X o si está en MsWindows directamente en el explorador de archivos escribiendo la dirección \\192.168.20.X.

5. Instalación del servidor DNS

El siguiente servicio que se pretende instalar es un servidor de nombres interno (DNS). La filosofía utilizada a lo largo de la asignatura consiste en realizar un despliegue de servicios mediante un servicio por máquina. Para no saturar los servidores de virtualización se propone realizar la instalación en la máquina VM2, junto con el servidor Samba. Concretamente se instalará el servidor BIND9 siendo este servicio interno para los equipos de la red virtual de cada alumno.

Para servir peticiones DNS procedentes de la red interna, el servidor DNS operará en modo DNS caché. En este modo el servidor consulta a servidores externos las direcciones IP y las guarda durante un período tiempo para no realizar más peticiones externas cuando se realice la misma petición.

Antes de instalar y configurar el servidor DNS en esta tarea debe familiarizarse con la herramienta *dig* encargada de mostrar la respuesta completa a peticiones DNS realizadas directamente a cualquier servidor DNS. La información mostrada por esta herramienta es muy detallada, por lo que debe analizar detenidamente la salida de cada ejemplo realizado en la siguiente tarea.

Tarea 11.- Instale el paquete *dnsutils* en todas las máquinas, necesario para revisar las respuestas de los servidores de nombres.

T11.1.- Desde la máquina GW haga una petición directa un servidor de nombres de la Universidad mediante el comando `dig @150.214.130.15 www.dte.us.es`. En la respuesta busque ANSWER SECTION con la respuesta ¿cual es la IP de www.dte.us.es?. Repita el proceso usando el servidor de nombres 192.168.20.3.

T11.2.- Intente resolver con alguno de los servidores anteriores el nombre noexiste.es. En la respuesta no encontrará ANSWER SECTION puesto que el dominio no existe, pero observe el texto tras *status*: ¿Cuál es el *status* para un dominio existente y para otro inexistente?

T11.3.- Ahora conecte a la máquina VM2 e intente resolver un nombre con los siguientes comandos: `dig @150.214.130.15 www.dte.us.es` y `dig @192.168.20.3 www.dte.us.es`. ¿Por qué ha fallado el segundo comando en VM2 y en cambio en GW sí funciona?.

T11.4.- Cambie la configuración del firewall de forma que, el único equipo de la red 192.168.7.0/24 con acceso a DNS sea VM2 y además, que sólo se tenga acceso a un único DNS externo: 192.168.20.3. No borre ninguna regla del fichero /etc/nftables.conf, sólo comente las líneas que no necesite y comente los cambios realizados en el mismo fichero para facilitar la evaluación.

T11.5.- Verifique el paso anterior comprobando que VM1/VM3 no es capaz de resolver nombres con ningún servidor externo, para ello use el comando *dig* haciendo peticiones directas "@" con un servidor de nombres externo desde la máquina VM1 o VM3, del mismo modo que se hizo en los primeros ejemplos de esta tarea.

Tras la configuración anterior del firewall los equipos de la red interna no pueden resolver nombres, excepto VM2. La solución propuesta consiste en instalar y configurar un DNS en VM2 que pueda ser usado por todos los equipos de la red interna y de esta forma evitamos suplantaciones DNS.

Tarea 12.- Se instalará un servicio DNS caché en la máquina VM2, instale en primer lugar el paquete *bind9* en esta máquina.

T12.1.- Edite el fichero /etc/bind/named.conf.options y busque la sección forwarders. Quite los comentarios a esta sección y usando la sintaxis correcta usando la siguiente IP como servidor externo: `192.168.20.3`.

T12.2.- Antes de continuar es necesario cambiar otra opción en este fichero de configuración para evitar multitud de problemas con los DNS de la Universidad. Concretamente busque en este fichero la directiva *dnssec-validation* y cambie su valor de auto a no, quedando establecida así: `dnssec-validation no;`

T12.3.- Ejecute el comando `named-checkconf` para comprobar si ha cometido errores en la configuración, los errores habituales son el olvido del ";" al final de cada sentencia. Si el comando anterior no muestra errores reinicie el servicio *bind9* mediante el comando `systemctl restart bind9` y compruebe si está operativo listando los procesos en ejecución que tienen abiertos puertos UDP (comando `ss -lunp`). ¿Cual es el nombre del programa con el que se ejecuta el servidor *bind*?

T12.4.- Haga una petición directa al servidor de nombres de VM2 con `dig @192.168.7.102 www.dte.us.es` y observe la repuesta. Repita el proceso desde VM1 y VM3 para verificar que siempre responde correctamente su nuevo DNS.

T12.5.- Si todas las respuestas son correctas, edite la configuración de todas las máquinas VM1, VM2 y VM3, para que utilicen únicamente el nuevo servidor de nombres, debe editar el fichero /etc/network/interfaces de las 3 máquinas. Tras el cambio se recomienda reiniciar cada máquina para después comprobar que funciona la resolución de nombres usando el comando host con el nombre de dominio interno .tai, por ejemplo el del profesor: `host profesor.tai`. ¿Cómo es posible que se resuelva este nombre si no existe en Internet?

5.1. Configuración de un DNS maestro

Para poder disponer de un dominio privado es necesario utilizar un DNS maestro. Dicho servidor será capaz de resolver tanto peticiones de nombres externos (modo servidor DNS caché), como internas. En nuestro caso, el dominio será el *uvus* del alumno con el sufijo *.tai*. En el ejemplo mostrado se usará por tanto *uvus.tai* y a lo largo del desarrollo del laboratorio debe utilizar sustituir la palabra *uvus* por su *uvus* real.

El objetivo es conseguir que el servidor de dominio responda con la IP correcta a los dominios mostrados en la tabla 4. Para lograrlo es necesario configurar el servidor DNS en modo maestro para el dominio *uvus.tai*.

Máquina	Dirección IP	Nombres de dominio
VM1	192.168.7.101	vm1.uvus.tai
VM2	192.168.7.102	vm2.uvus.tai ns.uvus.tai
GW	192.168.7.1	gw.uvus.tai
GW → VM1	192.168.20.X	uvus.tai www.uvus.tai

Tabla 4. Entradas DNS para el dominio uvus.tai.

Tarea 13.- La configuración de un DNS maestro se realiza mediante la edición de varios ficheros. El primer paso es añadir una nueva zona de búsqueda para nuestro dominio, en principio, sólo para búsquedas directas. Edite el fichero /etc/bind/named.conf.local y añada una nueva zona usando el dominio *uvus.tai*, para ello básese en el código 1.

```
// Dominio uvus.tai, cambielo por su nombre
zone "uvus.tai" {
        type master;                    // Tipo de servidor
        file "/etc/bind/db.uvus.tai";   // Fichero con los registros DNS
};
```

Código 1. Configuración DNS maestro para una zona.

T13.1.- En el paso anterior se ha especificado un fichero donde estarán las entradas DNS (db.uvus.tai) y este fichero no existe puesto que es el nuevo dominio. Debe crear un fichero vacío con este nombre en /etc/bind e incluir el contenido mostrado en el código 2, sustituyendo el dominio *uvus.tai* por el dominio correspondiente a su uvus. El formato de este fichero es un estándar definido en la RFC1035.

```
; Dominio de laboratorio uvus.tai
; En este fichero los comentarios comienzan con ';'

$TTL    1h                              ; Tiempo de expiracion por defecto

@       IN      SOA   ns.uvus.tai. root.uvus.tai. (
                              2023092900  ; Numero de serie: YYYYMMDDnn
                              2h          ; Tiempo de refresco
                              10m         ; Tiempo de reintento
                              1w          ; Tiempo de expiracion
                              1h  )       ; Tiempo de Cache TTL
;
@         IN      NS      ns              ; Servidor DNS -> ns.uvus.tai
@         IN      A       192.168.7.1     ; Host principal del dominio -> uvus.tai
@         IN      AAAA    ::1             ; IP version 6 (ipv6)

ns        IN      A       192.168.7.102   ; ns.uvus.tai
vm1       IN      A       192.168.7.101   ; vm1.uvus.tai
vm2       IN      A       192.168.7.102   ; vm2.uvus.tai
vm3       IN      A       192.168.7.103   ; vm3.uvus.tai
gw        IN      A       192.168.7.1     ; gw.uvus.tai
```

Código 2. Entradas DNS para el dominio uvus.tai.

T13.2.- Para revisar si la configuración es correcta hay que utilizar 2 comandos: *named-checkconf* y *named-checkzone*. Utilice el primer comando `named-checkconf` para comprobar si ha cometido errores al realizar la configuración. Si no hay errores debe usar el comando `named-checkzone uvus.tai /etc/bind/db.uvus.tai` pero adecuando los parámetros al nombre de zona que ha creado.

T13.3.- Reinicie el servicio mediante el comando `systemctl restart bind9`. Para comprobar si el servicio se ha iniciado correctamente utilice el comando `systemctl status bind9` y revise la bitácora del servicio mediante `journalctl -u named -e`. La última opción hace que se muestre el final de la bitácora, desplace el texto y verifique que ha cargado su zona correctamente.

T13.4.- Para comprobar si el servidor opera pruebe el comando `host` desde la propia máquina VM2 con los nombres *vm1.uvus.tai*, *vm2.uvus.tai* y *gw.uvus.tai*. Repita el proceso desde la máquina VM3 y VM1.

T13.5.- Use el comando `dig` con *uvus.tai* y ejecute el comando `ping` con *uvus.tai*,. ¿Qué IP resuelve?

Tarea 14.- Para conseguir una configuración real de un servidor DNS también es necesario configurar los resolución inversa.

T14.1.- Pruebe el comando *host* con una IP, por ejemplo `host 8.8.8.8`, si observa la salida en ella se indica el nombre asociado a esa IP. Use el comando `nslookup 150.214.141.196` y obtendrá también la resolución inversa ofrecida por los DNS para esta dirección IP pero con más información.

T14.2.- Pruebe el mismo comando con 192.168.7.101 y obtendrá un error ¿por qué?

T14.3.- Opcional-3e: Consulte documentación sobre cómo configurar BIND9 para la resolución inversa y termine la configuración de su servidor DNS de forma correcta, resolviendo las 4 direcciones IP internas a sus nombres de dominio creados en la tarea anterior.

5.2. Configuración de un DNS secundario

En las tareas anteriores se ha utilizado un DNS propio para esta asignatura ubicado en la máquina compartida (192.168.20.3). Este DNS está configurado para comportarse como DNS secundario de todos los dominios *.tai*. Los DNS secundarios necesitan obtener toda las entradas DNS del DNS maestro mediante una comunicación TCP en el puerto 53 llamada *Transferencia de Zona*. Esta operación consiste en transferir al completo la base de datos de zona creada desde el DNS maestro. Disponer del DNS secundario permitirá mantener operativo el servicio si falla el servidor maestro, en este caso, los secundarios responden con la última copia transferida desde su DNS maestro.

En el ejemplo concreto propuesto en este laboratorio, hará que todas las zonas de todos los alumnos sean visibles entre sí se puedan visitar las páginas Web de quienes tengan terminado el laboratorio. Por tanto, el objetivo es hacer público el servidor de DNS de forma segura, y para conseguirlo debe tener el siguiente comportamiento:

- Desde el DNS secundario (máquina compartida) se podrán realiza la transferencia de zona de nuestra zona DNS: *uvus.tai*.
- Desde cualquier equipo exterior se responderán peticiones DNS para resolver entradas de nuestra zona *uvus.tai*

- Cualquier petición externa de resolución de nombres que no sea de nuestra zona uvus.tai debe ser denegada.

El último punto es importante cuando se hace público un servidor DNS para evitar que lo utilicen como servidor DNS público. En la figura 4 me muestra un esquema con la configuración del filtrado de paquetes que hay que realizar en la máquina GW para conseguir lo descrito.

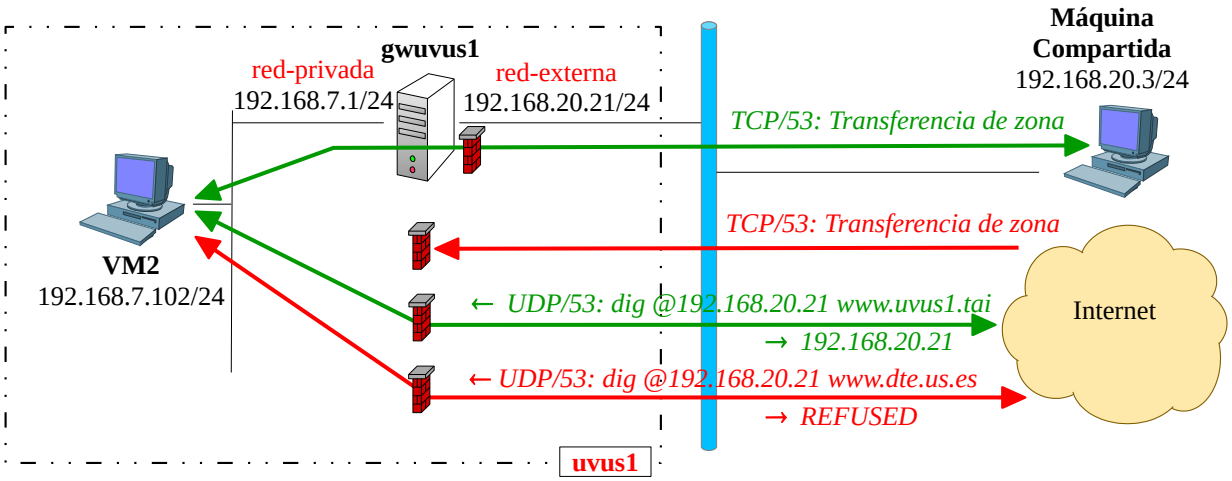


Figura 4. Configuración de netfilter para un DNS maestro.

Debe de seguir los pasos indicados a continuación pero usando los comandos adecuados en diferentes máquinas para asegurarse que el comportamiento al final es el indicado en la figura.

Tarea 15.- Para configurar el DNS secundario edite el fichero named.conf.local y añada las directivas indicadas en rojo para autorizar transferencias de la zona al DNS secundario.

```
// Dominio uvus.tai, cámbielo por su nombre
zone "uvus.tai" {
        type master;                    // Tipo de servidor
        file "/etc/bind/db.uvus.tai";   // Fichero con los registros DNS
        allow-transfer { 192.168.20.3; };
        also-notify { 192.168.20.3; };

};
```

Código 3. Configuración DNS secundario para una zona.

T15.1.- En el código 2 se estableció como servidor de nombres de su dominio a *ns.uvus.tai* el cual resuelve a la IP 192.168.7.102. Para hacer público su dominio, es necesario que el servidor NS sea una IP pública. Cambie la configuración del su zona para que *ns.uvus.tai* resuelva a su IP pública 192.168.20.X.

T15.2.- Ahora debe añadir el DNS secundario como entrada DNS de tipo NS para su dominio, pero las entradas tipo NS no admiten direcciones IP, sólo nombres. Use el comando `nslookup 192.168.20.3 192.168.20.3` para obtener el nombre de la IP 192.168.20.3 y añada una entrada de tipo NS a su dominio con el nombre obtenido, pero terminando en ".", es decir, debe poner un punto final adicional. ¿Por qué es necesario añadir este punto final?

T15.3.- Para que las transferencias tengan éxito hay que permitir en el firewall el tráfico TCP en el puerto 53, sólo entre la máquina VM2 y la IP del servidor secundario 192.168.20.3. ¿Son necesarias varias reglas DNAT?. Consulte por Internet como opera el protocolo DNS de transferencia de zonas para ver si el DNS maestro o el DNS secundario es quien comienza la comunicación.

T15.4.- Para comprobar la transferencia de zona hay que usar la opción axfr de *dig*. En VM3 y ejecute `dig axfr @192.168.7.102 uvus.tai` observando el error que aparece, debe denegar la transferencia de zona. Repita el comando desde la máquina compartida hacia su IP pública y así verificar que es posible transferir la zona completa. En caso de fallo debe revisar si ha configurado correctamente el firewall siguiendo el esquema de la figura 4.

T15.5.- Una vez resulto lo anterior, reinicie Bind9 con systemctl y revise el la bitácora con `journalctl -u named -e` buscando una línea que le indique que la zona *uvus.tai* ha sido transferida.

T15.6.- Para finalizar se revisará la seguridad del servidor DNS para que no se pueda abusar del mismo. Desde la máquina local use el comando `host microsoft.com 192.168.20.X` para comprobar que la respuesta es REFUSED. En cambio, si resuelve gw.uvus.tai debe obtener la respuesta correcta. Si no la obtiene y en ambos casos se agota el tiempo de espera, debe configurar adecuadamente el puerto UDP 53 según el esquema de la figura 4.

Cuando se modifica una zona es necesario notificar al servidor secundario que la zona ha cambiado y realizar una transferencia de zona. En este procedimiento entra en juego el número de serie establecido en la cabecera del fichero de definición de zona, es necesario incrementarlo para que se realice la transferencia.

Tarea 16.- Edite el fichero de zona creado en T13.1.-, cambie la dirección IP destino de *uvus.tai* a su IP pública e incremente el número de serie de la zona. Este último paso es necesario para disparar una transferencia y se actualice en el DNS secundario.

T16.1.- Ejecute en un terminal `journalctl -u named -f` y en otro terminal, también en VM2, fuerce una transferencia de zona mediante los comandos: `rndc reload uvus.tai` y `rndc notify uvus.tai`. En el primer terminal debe mostrar que se han realizado correctamente los comandos anteriores.

T16.2.- Para verificar la transferencia de zona anterior debe navegar desde la máquina compartida a http://uvus.tai y obtener la página Web alojada en su VM1. También deberá funcionar desde cualquier equipo de la red 192.168.20.0/24 que ya tenga terminada la Tarea 12.-

T16.3.- Compruebe que también se puede navegar a esa dirección desde VM3. En caso que no funcione debe cambiar las reglas DNAT del puerto 80 y 443 de forma que también se realice DNAT a las peticiones recibidas por la red interna.

5.3. Configuración multidominio

Este último ejercicio consistirá en configurar el servidor Web para servir páginas diferentes con varios nombres de dominios. Como colofón se realizará un ejercicio de suplantación (*web spoofing*) mediante la alteración de entradas de DNS y el servidor Web.

Tarea 17.- Usando el escritorio de la máquina VM3 inicie un navegador de Internet y navegue a http://vm1.uvus.tai/, http://192.168.7.101/. Observará que para ambas URLs se está navegando a la misma

máquina y se obtiene la misma página Web.

T17.1.- Navegue ahora a la raíz de su dominio, es decir a http://uvus.tai/. Pruebe también http://www.uvus.tai/ ¿que ocurre en este último caso?.

T17.2.- Añada una entada DNS en su dominio *uvus.tai* de forma que *www.uvus.tai* se resuelva con su dirección IP pública (192.168.20.X). Fíjese en el fichero mostrado en el código 2 para resolverlo

T17.3.- Asegúrese de incrementar el número de serie de la zona DNS para que el DNS secundario recargue la configuración. Después use en VM2 el comando `rndc reload uvus.tai` para forzar una recarga y avisar al DNS secundario.

Una vez resuelta la tarea anterior se dispone de 3 nombres de dominio diferentes apuntando a la misma dirección IP. Ahora el objetivo es servir páginas diferentes para cada uno de los nombres de dominio. Apache resuelve esta situación mediante la directiva *virtualhost*. Esta directiva abre un entorno de configuración donde se pueden definir diferentes comportamientos para cada nombre de dominio. Para cada nombre de dominio se pueden establecer diferentes opciones, principalmente será útil establecer el directorio donde están las páginas Web para cada uno de los dominios.

Tarea 18.- Como administrador en la máquina VM1 acceda al directorio /etc/apache2/sites-enabled. Liste los ficheros y copie el fichero 000-default.conf como uvus.conf. Edite este último fichero para que quede parecido al código 4, debe establecer correctamente todas las directivas resaltadas. En la configuración mostrada se indican los nombres de dominio *uvus.tai* y *www.uvus.tai* mediante las directivas *ServerName* y *ServerAlias*.

```
<VirtualHost *:80>
        ServerAdmin  webmaster@localhost
        DocumentRoot /var/www/uvus.tai
        ServerName   uvus.tai
        ServerAlias  www.uvus.tai
        ErrorLog     ${APACHE_LOG_DIR}/error.log
        CustomLog    ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

Código 4.Configuración Virtualhost para Apache2.

T18.1.- Una vez establecidos los nombres de dominio *Apache* muestra las páginas Web situadas en la carpeta indicada mediante la directiva *DocumentRoot*. En el código anterior este directorio está bajo /var/www, debe crear el nuevo directorio con el comando mkdir coincidiendo con la directiva *DocumentRoot* del código 4. Para el ejemplo mostrado sería `mkdir /var/www/uvus.tai`.

T18.2.- Cree un archivo index.html con el editor nano en el nuevo directorio creado en el paso anterior que contenga código HTML que muestre su nombre. Si no conoce HTML puede usar cualquier ejemplo de Internet y modificarlo.

T18.3.- Para verificar si la configuración es correcta ejecute el comando `apache2ctl configtest`, si no se muestra error puede reiniciar el servicio con `systemctl restart apache2`. Ahora, navegue a http://www.uvus.tai y http://uvus.tai para comprobar aparece su nombre en la página Web mostrada. Compruebe que en http://192.168.20.X aparece otra página, de forma que ya dispone de dos webs en el

mismo servidor.

T18.4.- Descargue desde Internet una plantilla HTML5 para sustituir la página que creó, puede descargarla de alguno de los enlaces mostrados a continuación. Compruebe que se muestra correctamente la plantilla en http://www.uvus.tai tras copiar correctamente los ficheros descargados.

- https://html5up.net
- https://startbootstrap.com
- https://getbootstrap.com/docs/5.3/examples/

T18.5.- Navegue a http://vm1.uvus.tai y verá otra página diferente ¿Donde está situada esta página? ¿Cual es el archivo de configuración que controla este nombre de dominio?

T18.6.- Para finalizar esta tarea y realizar la evaluación es necesario que aparezca su nombre, apellidos y el uvus en cada una de las Webs creadas. Edite los ficheros *index.html*, sin estropear las plantillas, de los diferentes sitios para que aparezca su nombre, apellidos y su uvus en algún lugar de la página Web de los siguientes sitios: http://www.uvus.tai, http://uvus.tai y http://vm1.uvus.tai.

T18.7.- Opcional-3f:Si navega mediante HTTPS a https://uvus.tai y https://www.uvus.tai observará que aparece la página predeterminada de Apache2. Configure Apache2 para que muestre la misma página que cuando se navega mediante HTTP.

El último ejercicio consistirá en realizar la suplantación de una página Web de la Universidad combinando la alteración de los DNS en la máquina VM2 con el servidor Web multidominio preparado en VM1. El objetivo será la plataforma de enseñanza virtual *ev.us.es*.

Tarea 19.- En el servidor de nombres del VM2 cree una nueva zona para *ev.us.es* del mismo modo que se hizo en la Tarea 13.-. Debe definir en esta zona la resolución para www.ev.us.es y ev.us.es, ambas apuntando a la dirección IP del VM1.

T19.1.- Compruebe si los nuevos dominios se resuelven correctamente desde la máquina VM3 usando los comandos `host` y `ping`.

T19.2.- Cree en la máquina VM1 una nueva configuración para estos dos dominios del mismo modo que se hizo en la Tarea 18.-. La página a servir está en el fichero *ev.zip* suministrado con el material del laboratorio. Debe descomprimirlo en una nueva carpeta dentro de `/var/www` llamada `ev.us.es` (`mkdir /var/www/ev.us.es`) y configurar correctamente Apache para que sirva esta página sólo para estos dos nombres de dominio.

T19.3.- Navegue desde la máquina VM3 a http://ev.us.es y compruebe el resultado.

6. Ejemplos de inseguridad en servicios Web

A continuación se propone la realización de dos ejercicios opcionales relacionados con la seguridad de los servidores Web y de los Navegadores. El primero consiste en realizar la suplantación de un sitio Web a pesar de tener SSL y el segundo en realizar la instalación de un *WebShell* en el servidor Apache.

Tarea 20.- Opcional-3g: Añada soporte SSL para el sitio *ev.us.es* en su servidor Apache utilizando como

certificados los ficheros suministrados ev.us.es.crt y ev.us.es.pem.

T20.1.- Navegue desde VM3 a https://ev.us.es y observe el error de seguridad que aparece, pero no añada una excepción para entrar, aunque puede que no le permita añadir la excepción.

T20.2.- El problema que presenta el certificado SSL de *ev.us.es* usado es que no se reconoce la autoridad certificadora que lo ha firmado. Para evitar este problema utilice el certificado de autoridad ca-tai.crt y añádalo como autoridad de confianza en la configuración de Firefox.

T20.3.- Vuelva a navegar a https://ev.us.es.

Tarea 21.- Opcional-3h: Añada el soporte del lenguaje PHP al servidor Web Apache instalando el paquete libapache2-mod-php.

T21.1.- Busque en GitHub algún *WebShell* escrito en PHP. Siga las instrucciones para que el WebShell quede empaquetado como un único fichero PHP con el nombre lista-notas.php.

T21.2.- Añada dicho fichero a su servidor en la siguiente dirección http://192.168.20.X/lista-notas.php y compruebe el nivel de acceso que tiene a su servidor.

Tarea 22.- Opcional-3i: Termine el ejercicio de Web spoofing de las tareas: Tarea 19.- y Tarea 20.- de forma que en vez aparecer la bandera pirata robe las credenciales de acceso intentando hacer una copia de la página de inicio de sesión usada por la Universidad. Como guía debe seguir los siguiente pasos

T22.1.- Intente descargar el HTML de la página de inicio de sesión de la Universidad https://sso.us.es/ que aparece tras pulsar sobre el inicio de sesión de en ev.us.es

T22.2.- Este HTML debe ponerlo dentro del propio servidor Web con algún otro nombre como parecido como *sso.html* o incluso *sso.php* si quiere ejecutar código en el servidor. ¿Consigue que se vea parecida la página falsa de inicio de sesión en el navegador?. Para conseguirlo deberá dedicarle tiempo.

T22.3.- Debe plantearse como realizar la captura de los datos del formulario y como alterar el comportamiento del botón Aceptar para que guarde o envíe los datos capturados.

T22.4.- El botón Aceptar debe capturar los datos, pero una vez pulsado, el usuario debe ser redirigido a la página real de inicio de sesión intentando disimular lo que ha ocurrido.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 4 - VNPs

Enunciados de laboratorios
Tecnologías Avanzadas de la Información
Rev. 2 - 30/11/23

1. Introducción y objetivos

La seguridad en las comunicaciones en redes IP puede ser abordada en diferentes capas: nivel de enlace, nivel IP, nivel de aplicación. Independientemente de la capa, el uso del cifrado de datos es la base de la seguridad usándose tanto cifrado simétrico como asimétrico según la tecnología involucrada. Este laboratorio está orientado al uso de tecnologías VPN que son fundamentales para interconectar redes privadas a través de redes públicas de forma segura. La duración estimada de esta sesión de laboratorio es de **4 horas**.

En la teoría de esta asignatura se tratan ampliamente las VPNs y las técnicas de cifrado en las que se basan estas tecnologías y, para profundizar en los conocimientos teóricos, se propone en este laboratorio usar primero IPSec en modo transporte para asegurar la red local. Este último ejemplo no se considera una VPN, por ello, después sí se usarán dos tipos de VPNs: una centralizada (OpenVPN) y otra distribuida (TINC).

Respecto a IPSec, se puede resumir como un conjunto de protocolos estandarizados de nivel de IP que permite asegurar la comunicación entre dos equipos y que además, permite crear VPNs cuando trabaja en modo túnel. En cambio, tanto OpenVPN como TINC son soluciones VPNs software de código abierto basadas en SSL/TLS. Las dos últimas son más sencillas de usar que IPSec y cubren un amplio rango de aplicaciones como: acceso remoto, unión de nodos remotos mediante VPN, seguridad Wi-Fi, balanceo de carga, etc. Su principal ventaja frente a otras soluciones comerciales es la facilidad y reducido coste de implantación. Ambas tecnologías operan en la capa 2 o 3 del modelo OSI uniendo, mediante túneles, todos los nodos distribuidos por la red. Se requiere una instalación tanto en el cliente como en el servidor y son compatibles con Linux, Windows, MacOS, Android y algunos más.

En esta sesión de laboratorio el alumno debe poner operativo IPSec y dos VPNs en el entorno virtual de laboratorio. En las primeras secciones se realiza de manera guiada la instalación y la puesta en marcha básica. Posteriormente se plantean tareas de mayor dificultad para realizar de manera no guiada, no siendo

necesaria la realización de las últimas propuestas de configuración.

Para facilitar el desarrollo de la sesión de laboratorio se propone la instalación de los paquetes indicados en la tabla 1, donde se indica la máquina donde debe instalarlo.

Nombre del paquete	Descripción	Ubicación
strongswan	Implementación OpenSource de IPSec	VM1, VM2 y GW
xca	Utilidad gráfica para gestión de certificados digitales	Máquina local o máquina VM3 (requiere entorno gráfico)
openvpn	OpenVPN	Todas la máquinas
tinc	TINC	VM1, VM2 y VM3
mc	Recomendación: Administrador de archivos para consola de texto	Todas las máquinas

Tabla 1. Programas necesarios para la realización de la sesión de laboratorio.

2. IPSec

IPSec es un conjunto de protocolos IP (capa 3) orientados a dotar de seguridad las comunicaciones entre dos equipos. El objetivo de IPSec es conseguir autenticidad, integridad y confidencialidad a los protocolos de superiores a IP pudiendo funcionar en dos modos: transporte o túnel. En este laboratorio se pretende dotar de seguridad a la red local interna existente entre GW, VM1, VM2 y VM3 (192.168.7.0/24) usando IPSec en modo transporte.

El modo transporte de IPSec ya se ha estudiado en la teoría de la asignatura por lo que no se presenta en profundidad. De forma resumida, cuando IPSec trabaja en este modo, la cabecera de la capa 3 no se modifica ni se incluye en la carga cifrada, es decir, sólo se cifra la carga del protocolo IP. Este modo es útil para cifrar las comunicaciones entre 2 puntos, pero se complica si se quieren asegurar las comunicaciones IP entre multitud de equipos ya que hay que configurar los equipos por pares.

Tarea 1.- Instale el paquete *tcpdump* en VM1 y GW para proceder realizar una captura de datos de una sesión FTP de la siguiente forma:[1]

- En VM1 ejecute el comando `tcpdump -A port ftp > dump.out` como administrador. Este comando capturará todos los datos del puerto 21 y los volcará en el fichero dump.out.
- Desde la red externa inicie una sesión con el comando *ftp* hacia si IP pública usando el usuario *web* y su contraseña, liste los ficheros y salga de la sesión *ftp* con `exit`.
- Pulse CTRL+C en las capturas y con el editor de textos o el comando *less* compruebe si en el fichero dump.out se ha capturado el inicio de sesión un la contraseña, tiene que buscar la cadena PASS.

En el ejemplo anterior se presentó la facilidad de capturar los inicios de sesiones de protocolos no cifrados

[1] Si ha realizado la tarea opcional del laboratorio anterior activando TLS en FTP, debe quitar esta característica para realizar esta tarea.

como FTP simplemente capturando paquetes de la red. Muchos de estos protocolos incluyen algún modo para activar TLS y asegurar el protocolo, pero otros, no lo contemplan delegando la seguridad en capas inferiores al nivel de aplicación. Para este último escenario una solución bastante simple y fácil de configurar es IPSec en modo transporte.

IPSec en Linux está soportado a nivel de núcleo pero es necesario añadir un software extra de nivel de usuario para interactuar con el API del núcleo ya que se deben negociar y establecer las claves de cifrado entre los equipos involucrados. Para este fin, *StrongSwan* es un software de código abierto para la configuración de IPSec y está disponible para Linux y Android.

Tarea 2.- Instale el paquete *strongswan* en VM1 y en GW.

T2.1.- Edite el fichero `/etc/ipsec.conf` de VM1 y para configurar la pareja de equipos VM1-GW añada la conexión indicada a continuación:

```
conn tai-seguro
  type = transport
  authby = secret
  left = 192.168.7.101
  right = 192.168.7.1
  pfs = yes
  auto = start
```

Código 1. Ejemplo de conexión IPSec.

T2.2.- Para la configuración anterior es necesario establecer una clave compartida en `/etc/ipsec.secrets`. Edite este fichero y establezca una clave para la pareja VM1-GW siguiendo el siguiente formato:

```
192.168.7.101 192.168.7.1 : PSK "clave-secreta"
```

T2.3.- Esta configuración tiene que ser duplicada en GW pero intercambiando las IPs en los dos ficheros. Una vez configurado GW inicie el servicio con `systemctl start ipsec` en ambas máquinas y mediante el comando `ipsec status` verifique que aparece la conexión establecida. Si no aparece nada debe buscar en la bitácora el error, la puede revisar con `journalctl -e` o `less /var/log/syslog`.

T2.4.- Si la conexión IPSec está establecida, cualquier paquete TCP/IP y UDP/IP será cifrado en un paquete IPSec usando el protocolo IP 50 (ESP). Verifique que esto ocurre usando `tcpdump proto 50` en un terminal de VM1 y ejecutando `ftp 192.168.7.101` desde GW.

T2.5.- Repita la Tarea 1.- para comprobar que ya no es posible capturar paquetes en el puerto 21 provenientes de GW en cambio, si inicia la sesión desde VM3 sí se podrán capturar.

T2.6.- Opcional-4a: Añada IPSec a toda la red privada añadiendo VM2 y VM3. ¿Cuantas conexiones tiene que configurar?

3. Instalación de OpenVPN

La instalación de OpenVPN se realiza de manera similar a la de cualquier aplicación en Debian. La mayoría de los paquetes en Debian tras la instalación contienen una configuración mínima para la puesta en marcha del servicio correspondiente, como puede ser, el servidor SSH. En cambio, la puesta en funcionamiento de OpenVPN requiere una configuración manual, especificando para cada máquina el perfil que tendrá: servidor VPN, cliente VPN o ambos. Por ello el sistema de paquetes Debian no incluye una configuración predeterminada, pero facilita la configuración con una serie de ejemplos de configuración incluidos en la documentación del paquete.

A continuación se instalará OpenVPN, y tras la instalación se indicará cómo ubicar correctamente los archivos de configuración en el directorio /etc/openvpn. Estas instrucciones son necesarias ya que OpenVPN no trae establecida ninguna configuración predeterminada. El directorio /usr/share/doc/openvpn incluye documentación y ejemplos para la puesta en funcionamiento que deberá usar a lo largo de las siguientes tareas.

Tarea 3.- Instale el paquete *openvpn* en la máquina GW, como se ha comentado no trae ninguna configuración predeterminada, por lo que no se activará ningún servicio en la máquina.

T3.1.- Acceda a la carpeta /usr/share/doc/openvpn y encontrará toda la documentación y ejemplos disponibles para la configuración. Entrando en el directorio examples encontrará un directorio llamado sample-config-files con ejemplos para utilizar en la configuración tanto del cliente como el servidor.

T3.2.- Antes de la puesta en funcionamiento de OpenVPN será necesario crear los certificados digitales necesarios para que opere correctamente, llegados a este punto existen dos posibilidades:

1. La opción recomendada en esta sesión de laboratorio es el uso de XCA, ya que mostrará de una forma más clara como se están creando los certificados y es una herramienta con interfaz gráfica.
2. La opción rápida (no recomendada) es utilizar easy-rsa. Éste último es un software para generar y mantener una base de datos con los certificados de la VPN, pero funciona en la consola de comandos. Si usa esta opción, los pasos indicados en el howto de OpenVPN en la siguiente dirección: https://openvpn.net/community-resources/how-to. Si elige esta opción obvie la Tarea 4.- y la Tarea 5.-

3.1. Generación de los certificados digitales

OpenVPN opera utilizando certificados digitales en cada nodo de la VPN, incluido el propio servidor. Para asegurar la red VPN y administrar los certificados se requiere una autoridad de certificación encargada de firmar todos los certificados y con posibilidad de incluir una lista de revocación (CRL). Todos estos componentes forman una infraestructura de clave pública (PKI) utilizada para operar en la VPN.

La fortaleza de la seguridad en la VPN recae en la clave privada de la autoridad de certificación, por ello debe mantenerse en lugar seguro. Los requerimientos de claves y certificados se pueden resumir como sigue:

- El administrador de la VPN crea su clave pública, su clave privada y un certificado digital que incluye su clave pública. Actuará como autoridad de certificación, por tanto, este certificado estará firmado por él mismo (autofirmado). El administrador debe mantener la clave privada de la autoridad de

certificación en secreto.

- Para el servidor se crea una clave pública, una privada y un certificado que incluye la clave pública. La autoridad de certificación firma el certificado digital del servidor con su clave privada.
- Cada cliente de la VPN se genera su clave pública y su clave privada. La autoridad firma, para cada cliente, un certificado digital que incluye la clave pública del cliente en cuestión.
- El administrador publica para todos los clientes/servidores la clave pública de la autoridad en un certificado, para que así puedan validar todas las firmas digitales.

En la tabla 2 se muestran los componentes requeridos en cada uno de los equipos conectados a OpenVPN. Se puede resumir que cada equipo (incluido el servidor) dispondrá de tres claves: la clave pública de la autoridad de certificación, su propia clave pública y su clave privada. Además de los ficheros indicados, el servidor requiere unos parámetros extra, necesarios para realizar un intercambio seguro de claves utilizando el protocolo de intercambio de claves de *Diffie-Hellman*.

Contenido	Ubicación	Secreto
Clave privada de la autoridad de certificación	Ubicación de máxima seguridad, incluso offline en un medio extraible.	**Sí**, sólo accesible al administrador de la VPN
Certificado de la autoridad (incluye la clave pública)	En todos los clientes y el servidor	**No**
Clave privada del servidor VPN	En el servidor con los permisos adecuados para evitar su copia	**Sí**, sólo accesible al servidor
Certificado firmado con la clave pública del servidor VPN	En el servidor	**No**
Clave privada de cada cliente	Cada clave sólo en el cliente correspondiente con los permisos adecuados para evitar su copia	**Sí**, sólo accesible para el cliente
Certificado firmado con la clave pública de cada cliente	Cada certificado sólo en el cliente correspondiente	**No**

Tabla 2. Ubicación de los certificados firmados y claves privadas necesarias.

Todos estos elementos se generarán a continuación con la herramienta XCA. Con esta herramienta se pueden realizar todos los pasos indicados anteriormente. Para generar los certificados digitales para el servidor y clientes en OpenVPN hay ciertos requerimientos que deben cumplir que se resumen en la tabla 3. En este sentido, para que OpenVPN opere correctamente siga los pasos de la siguiente tarea cuidadosamente.

Nodo	Key usage	Extended key usage
Cliente	digitalSignature	TLS Web Client Authentication
	keyAgreement	

	digitalSignature, keyAgreement	
Servidor	digitalSignature, keyEncipherment	TLS Web Server Authentication
	digitalSignature, keyAgreement	

Tabla 3. Requerimientos para los campos de los certificados en OpenVPN.

Tarea 4.- Instale el paquete *xca* desde el gestor de paquetes en VM3 e inicie el programa. Se utilizará este programa para crear los certificados de la siguiente forma:

T4.1.- Inicie el programa *xca* y cree una nueva base de datos para almacenar sus certificados.

T4.2.- Cree una nueva autoridad de certificación encargada de firmar todos los certificados de los clientes que se conectarán a la VPN. Para ello, acceda a la pestaña Certificados y pulse el botón Nuevo certificado. Utilice la figura 1 para configurar correctamente las dos primeras pestañas:

- En la pestaña Origen debe seleccionar la plantilla [default] CA y pulsar en Aplicar todo.
- Cada alumno debe rellenar con su nombre y apellidos el campo organizationalUnitName.
- En el campo commonName debe poner: *autoridad*.
- El correo electrónico debe estar en el campo emailAddress de la forma: uvus@alum.us.es
- En el resto de pestañas deje con los valores predeterminados.

T4.3.- El siguiente paso es crear un certificado para el servidor. Para conseguirlo, cree un nuevo certificado y configure las diferentes opciones según se indica en la figura 2. Siga la figura 2 correctamente considerando los siguientes requerimientos:

- El certificado debe estar firmado por la autoridad certificadora.
- En el campo commonName debe poner: *servidor*.
- El campo de uso de clave (*Key Usage*) debe contener *Digital Signature* y *Key Encipherment*
- El campo extendido de uso (*Extended Key Usage*) debe contener *TLS Web Server Authentication*.

T4.4.- En el caso de los clientes se debe repetir el proceso pero con otras opciones. Debe crear un certificado para el cliente firmado por la autoridad certificadora para VM1. La configuración se muestra en la figura 3 y los requerimientos son los siguientes:

- El certificado debe estar firmado por la autoridad certificadora.
- En el campo commonName debe poner: *cliente1*.
- El campo de uso de clave (*Key Usage*) debe contener *Digital Signature* y *Key Agreement*
- El campo extendido de uso (*Extended Key Usage*) debe contener *TLS Web Client Authentication*.

La herramienta XCA mantiene todos los datos sobre los certificados en un único fichero que actúa como base de datos. Ahora se deben extraer las diferentes claves privadas y certificados en diferentes ficheros para incluirlos en la configuración de OpenVPN.

Tarea 5.- Se procederá a la exportación de los certificados y de las claves privadas de la siguiente forma:

T5.1.- Se exportarán las claves privadas del *servidor* y del *cliente1*. No debe exportar la clave privada de la autoridad, ésta sólo se utiliza para el firmado y debe permanecer bajo un máximo nivel de seguridad. Para ello acceda a la pestaña Private Keys y seleccione la clave del cliente1 y del servidor, después use el botón Export. Use las opciones predeterminadas en la exportación de los ficheros, se generarán ficheros con extensión *.pem.*

T5.2.- Ahora se procederá a la exportación de los certificados desde la pestaña Certificates. Seleccione el certificado de la *autoridad*, el del *servidor* y el *cliente1*, y expórtelos con las opciones por defecto. Obtendrá diferentes ficheros con extensión *.crt* con las claves públicas.

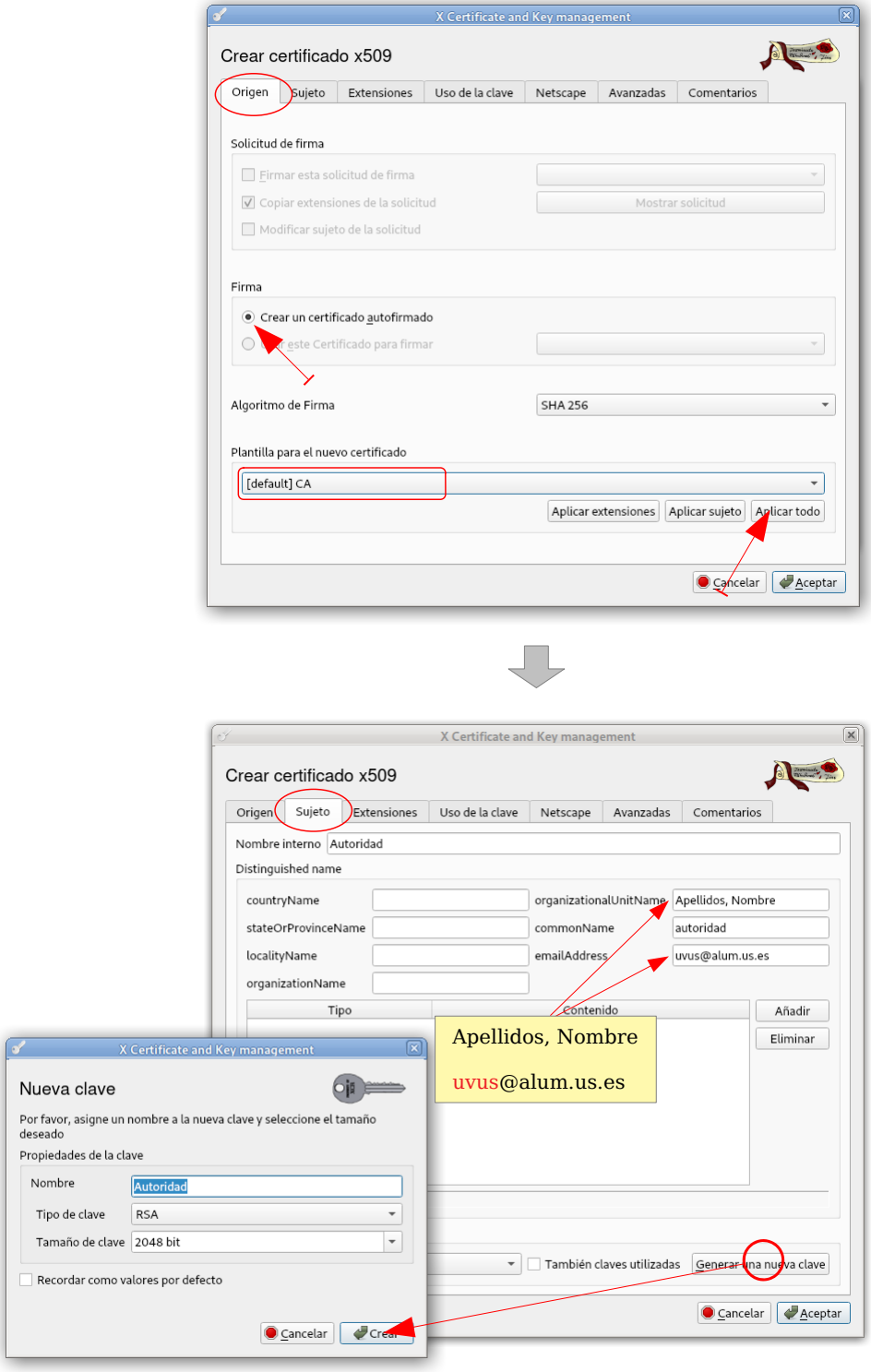


Figura 1. Generación de la autoridad de certificación.

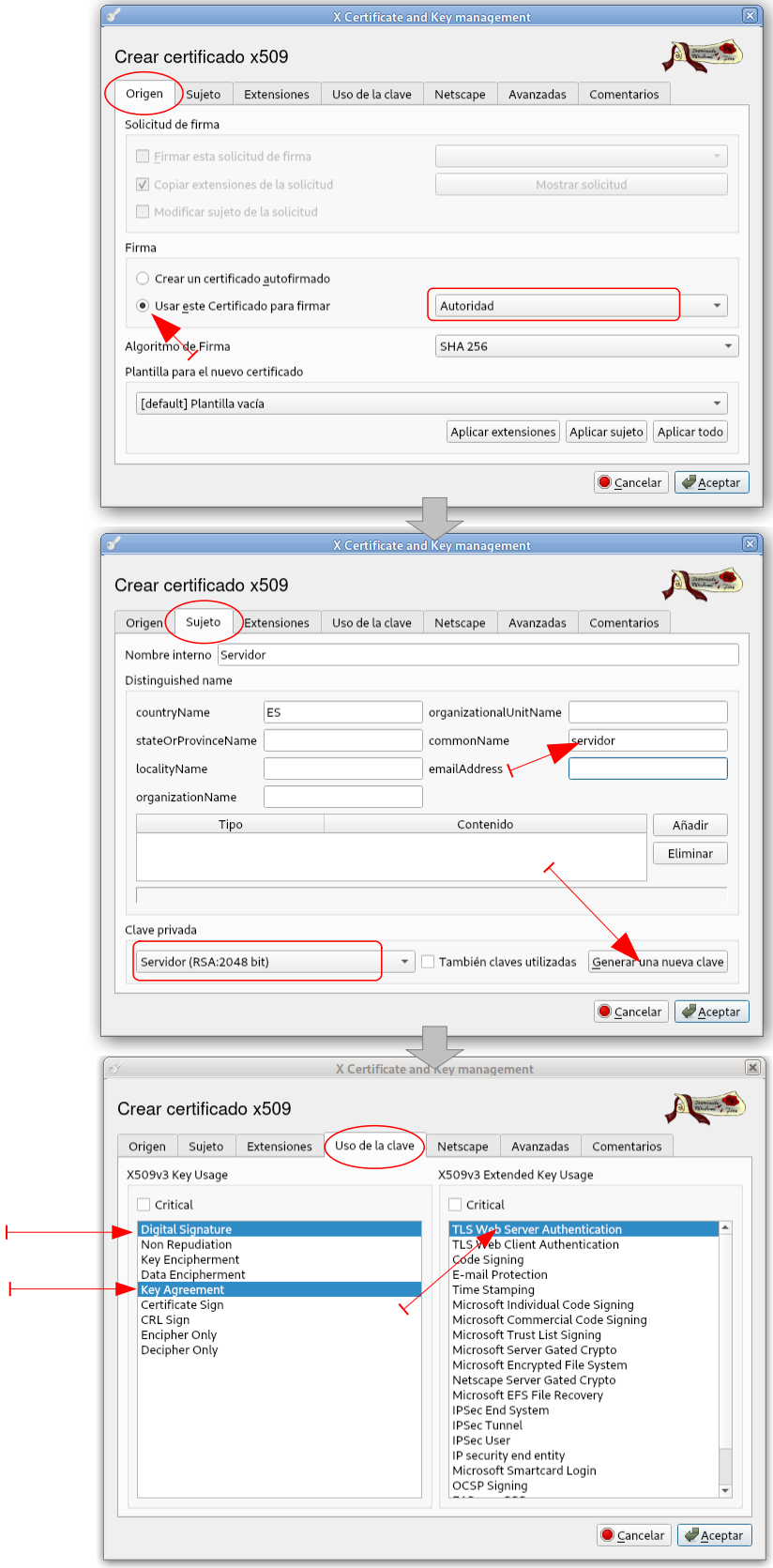


Figura 2. Generación del certificado del servidor.

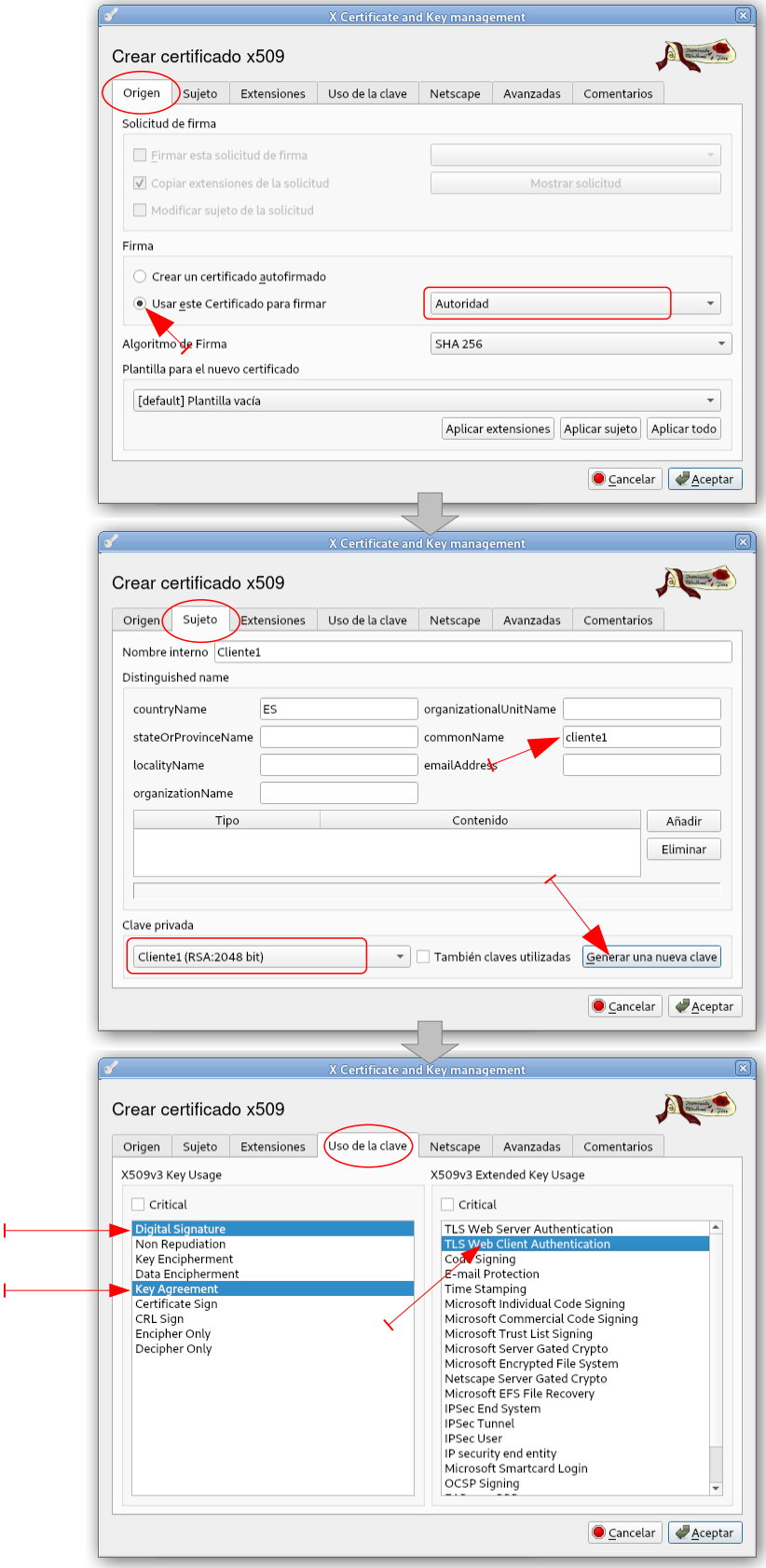


Figura 3. Generación del certificado del cliente.

3.2. Configuración del servidor y los clientes

Los ficheros obtenidos anteriormente se deben distribuir adecuadamente entre los clientes y el servidor OpenVPN. Además, hay que asegurarse de proteger adecuadamente aquellos que contienen las claves privadas, sólo deberían ser accesibles por el programa OpenVPN o el usuario root en su caso, en cada uno de los equipos.

Se procederá a la configuración del servidor y de los clientes, utilice la figura 4 para tener una visión global de cómo deben quedar los certificados y claves en cada una de los equipos virtuales.

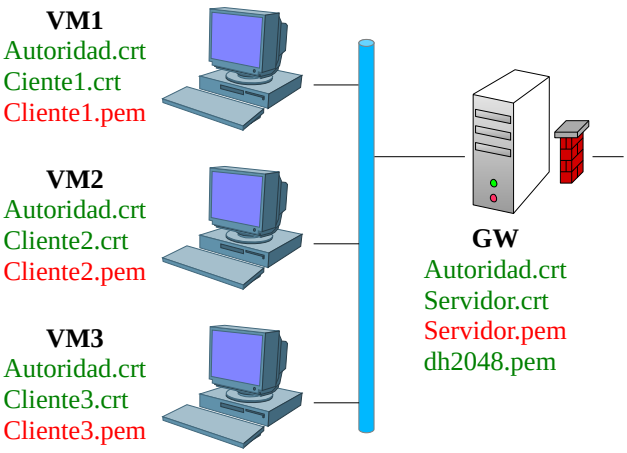


Figura 4. Ubicación de certificados y claves en la VPN.

Tarea 6.- Para comenzar la configuración del servidor asegúrese de tener instalado el paquete *openvpn.*

T6.1.- La configuración del servidor requiere un fichero adicional llamado parámetros de *Diffie-Hellman* utilizados para el intercambio seguro de claves privadas. Desde XCA genere este fichero usando el menú Extra → Generate DH parameter y estableciendo los bits a 2048. Obtendrá un fichero necesario sólo en el servidor.

T6.2.- En el servidor (GW) debe copiar 4 ficheros en la ubicación /etc/openvpn/ usando el comando *scp*. Los ficheros son los indicados en la tabla 4 o en la figura 4 y los nombres pueden variar según como los hubiera guardado de la tarea anterior.

Fichero	Contenido
Autoridad.crt	Certificado de la autoridad certificadora con su clave pública
dh2048.pem	Parámetros Diffie-Hellman generados en T6.1.-
Servidor.crt	Certificado del servidor con la clave pública firmado por la autoridad de certificación
Servidor.pem	Clave privada del servidor

Tabla 4. Ficheros necesarios en el servidor OpenVPN.

T6.3.- Una vez copiados en /etc/openvpn, como administrador liste el directorio /etc/openvpn con el comando `ls -l`. Debe cambiar los permisos del todos los ficheros asegurándose que pertenezcan al usuario *root* y que sólo el usuario *root* tenga permiso de lectura sobre el fichero con la clave privada del servidor, el comando `ls -l /etc/openvpn` debería mostrar la siguiente salida:

```
-rw-r--r-- 1 root root 1513 nov 24 20:21 Autoridad.crt
drwxr-xr-x 2 root root 4096 feb 20  2019 client
-rw-r--r-- 1 root root  424 nov 24 20:29 dh2048.pem
drwxr-xr-x 2 root root 4096 feb 20  2019 server
-rw-r--r-- 1 root root 1237 nov 24 20:26 Servidor.crt
-rw------- 1 root root 1675 nov 24 20:20 Servidor.pem
-rwxr-xr-x 1 root root 1468 feb 20  2019 update-resolv-conf
```

T6.4.- Copie el fichero /usr/share/doc/openvpn/examples/sample-config-files/server.conf que contiene una configuración de ejemplo a /etc/openvpn.

T6.5.- Ahora debe editar el fichero de configuración /etc/openvpn/server.conf y establecer los nombres de los ficheros de certificados a los nombres adecuados cambiando las líneas indicadas a continuación:

```
# Any X509 key management system can be used.
ca Autoridad.crt
cert Servidor.crt
key Servidor.pem  # This file should be kept secret

# Diffie hellman parameters.
dh dh2048.pem

# Network topology
topology subnet

# ¡Comente esta opción!
# tls-auth ta.key 0
```

Código 2. Cambios en el fichero de configuración para el servidor OpenVPN.

T6.6.- Ejecute el comando `openvpn server.conf` para iniciar el servidor desde la línea de comandos. Este inicio es temporal para comprobar que no existen errores de configuración. Para parar el servidor pulse CTRL+C.

T6.7.- Inicie el servicio OpenVPN mediante el comando `systemctl start openvpn@server`. Si todo ha ido bien, con el comando `ip a` aparecerá una nueva interfaz llamada tun0 con una IP asignada. Ésta es la red interna de la VPN, perteneciendo esta IP al servidor. Si no inicia correctamente, use el comando `systemctl status openvpn@server` para leer los mensajes de error y corregirlos.

T6.8.- En un terminal visualice la bitácora del sistema con el comando `journalctl -f -u openvpn@server`, podrá ver los cambios en tiempo real mientras trabaja en el resto de tareas. Manteniendo este terminal así, le ayudará a detectar problemas cuando otros clientes se conecten al servidor.

Para continuar con la configuración hay que añadir los clientes en la VPN. pero sólo se configurará inicialmente VM1 para detectar posibles errores y después las demás. Cada uno de los clientes necesitará tres ficheros para poder unirse a la VPN, siendo estos clientes la máquinas VM1, VM2 y VM3

- Certificado de la autoridad certificadora con su clave pública: *Autoridad.crt*
- Certificado del cliente con la clave pública y firmado por la autoridad: *ClienteX.crt*

- Clave privada del cliente: *ClienteX.pem*

Tarea 7.- En primer lugar se va a configurar el *cliente1* en VM1 siguiendo los siguientes pasos:

T7.1.- Instale con APT paquete *openvpn* en VM1.

T7.2.- Copie el fichero de configuración de ejemplo para los clientes desde la ubicación `/usr/share/doc/openvpn/examples/sample-config-files/client.conf` en el directorio `/etc/openvpn`.

T7.3.- Cada cliente necesita sólo los ficheros indicados tabla 4 / figura 4, deberá transferirlos por SSH y para ello puede utilizar el comando *scp*, el administrador de ficheros del escritorio de la máquina VM3 o el programa *mc*. Transfiera los tres ficheros necesarios a la ubicación `/etc/openvpn` del cliente.

T7.4.- Establezca los permisos de los ficheros del directorio `/etc/openvpn` adecuadamente como se hizo en T6.3.-.

T7.5.- Edite el fichero `/etc/openvpn/client.conf` estableciendo los valores correctos para los certificados y claves tal y como se mostró en el código 2. Considere que ahora los dos últimos ficheros tienen nombres diferentes.

T7.6.- También en el mismo fichero de configuración debe establecer la dirección IP del servidor VPN, busque en este fichero la directiva `remote` para establecerla a `remote 192.168.7.1 1194`. Además debe comentar la directiva `tls-auth`.

T7.7.- Inicie OpenVPN en una consola mediante el comando `openvpn client.conf` para ver si inicia correctamente o muestra mensajes de error. Este inicio es temporal para comprobar que no existen errores de configuración.

T7.8.- Sin parar la ejecución del paso anterior, use una nueva consola ejecutando el comando `ip a` para comprobar si aparece la interfaz `tun0` y tiene una IP asignada. En caso contrario puede tener mal generados los certificados o la configuración.

T7.9.- Si el paso anterior fue correcto, pare la consola donde se está ejecutando OpenVPN usando CTRL+C e inicie el servicio de manera correcta `systemctl start openvpn@client`. Si no inicia, use el comando `systemctl status openvpn@client` para solucionar el problema.

T7.10.- Acceda a la bitácora del servicio OpenVPN para ver los mensajes de información y/o error tanto en el cliente como en el servidor con el comando `journalctl -u openvpn@server` (en GW) y `journalctl -u openvpn@client` (en VM1). Debe interpretar el resultado de la negociación entre el cliente y el servidor de la VPN.

T7.11.- Como última comprobación, debe reiniciar todas las máquinas y comprobar que el servicio OpenVPN se inicia automáticamente. Así, reinicie las 3 máquinas y mediante `systemctl status ...` y e `ip a` compruebe el estado del servicio en cada máquina y las IPs asignadas en la interfaz `tun0` respectivamente.

Tarea 8.- Añada las máquinas VM2 y VM3 a la VPN generando nuevos certificados y repitiendo los pasos de la tarea anterior.

Tarea 9.- Entre en el directorio `/var/log/openvpn/` del servidor VPN y liste los ficheros. Visualice el contenido del fichero `ipp.txt` y `openvpn-status.log`. Interpretando correctamente el contenido, realice lo

siguiente:

T9.1.- Ejecute el comando *ping* desde el servidor a los clientes a través de la VPN.

T9.2.- Ejecute el comando *ping* desde un cliente a otro cliente a través de la VPN.

3.3. Revocación de certificados

Una situación habitual en la administración de cualquier VPN es la expulsión de equipos de la red. Independientemente del motivo (claves comprometidas, bajas de equipos, etc.), OpenVPN implementa este procedimiento usando una lista de revocación de certificados (CRL en inglés). Un CRL es una parte esencial de un PKI y no es más que la lista de certificados revocados firmado por la autoridad de certificación. Desde XCA generar el listado CRL es fácil. En el siguiente ejemplo revocaremos el certificado de la segunda máquina virtual y se configurará el servidor para operar correctamente con el CRL.

Tarea 10.- Inicie XCA y revoque el certificado de la máquina virtual VM2. Se puede revocar seleccionando el certificado y usando el menú flotante que aparece con el botón derecho del ratón sobre el certificado.

T10.1.- Ahora seleccione el certificado de la autoridad de certificación y vuelva a desplegar el menú con el botón derecho. Aparecerá un submenú CA → Generate CRL.

T10.2.- En la pestaña Revocation lists aparecerá la lista recién creada, use el botón exportar para generar el fichero CRL, establezca el nombre del fichero a *AutoridadCRL.pem*.

Tarea 11.- El último paso es cambiar la configuración del servidor OpenVPN.

T11.1.- Copie el fichero CRL llamado *AutoridadCRL.pem* en el directorio /etc/openvpn del servidor.

T11.2.- Edite la configuración del servidor (fichero /etc/openvpn/server.conf) y añada la directiva indicada en el código 3 con el nombre del fichero CRL.

```
ca Autoridad.crt
cert Servidor.crt
key Servidor.pem
crl-verify AutoridadCRL.pem  # Este es fichero contiene la lista de certificados
                             # revocados
```

Código 3. Configuración del listado CRL en OpenVPN.

T11.3.- Reinicie el servicio OpenVPN del servidor con `systemctl restart openvpn@server` y utilice el comando `journalctl -f -u openvpn@server` para observar el comportamiento del servidor. Debe reiniciar los clientes ya que tardan cierto tiempo en reconectar al servidor.

T11.4.- ¿Observa el aviso en el servidor sobre el cliente con el certificado revocado? ¿Qué ocurriría si se revoca el certificado del servidor?

3.4. Modos de funcionamiento de OpenVPN (opcional)

Las siguientes tareas se consideran opcionales ya que algunas presentan cierta dificultad. Se propone que tras conseguir el funcionamiento básico de OpenVPN con las tareas anteriores, se pruebe ahora diversas

configuraciones y modos de funcionamiento avanzados. Si lee detenidamente los comentarios del fichero de configuración del servidor, observará multitud de ejemplos y directivas comentadas que alteran el funcionamiento predeterminado de OpenVPN, tanto en los clientes como en el servidor.

Las siguientes tareas debe resolverlas cambiando los ficheros configuración del servidor y los clientes según los ejemplos indicados en los propios ficheros de configuración. Haga una copia de seguridad de los ficheros de configuración antes de afrontar las siguientes tareas.

Tarea 12.- Opcional-4b: ¿Cómo está funcionando su VPN usando UDP o TCP? Averígüelo y cambie el modo de funcionamiento. Use el comando `ss` para localizar el puerto donde está el servidor.

> **T12.1.-** En la configuración realizada de OpenVPN se comentó la directiva `tls-auth`, búsquela en el archivo de configuración y lea el motivo por el cual no se debe comentar. ¿Tras el cambio, en la tarea Tarea 12.- es necesaria activarla?
>
> **T12.2.-** Debe activar esta opción en toda la VPN y conseguir que opere correctamente siguiendo las instrucciones de la página oficial de OpenVPN.

Tarea 13.- Opcional-4c: Establezca una IP fija para cada cliente, para ello mire el ejemplo indicado donde se utilizan ficheros ubicados en un directorio llamado *ccd*. Debe establecer para VM1 la IP 10.8.0.101, para VM2 la IP 10.8.0.102 y para VM3 la IP 10.8.0.103.

4. TINC

TINC es una solución software para la creación de VNPs distribuidas (mesh). Este software es capaz de autoconfigurar las rutas de manera dinámica para que todos los clientes de la VPN sean accesibles entre sí. TINC distingue dos tipos de nodos, aquellos que permiten a otros nodos que se conecten a él (nodos maestros) y nodos clientes. La principal diferencia con OpenVPN es la existencia de tantos nodos de tipo servidor como se deseen, así la red queda asegurada ante posibles desconexiones de servidores. Otra característica de este tipo de VPN es que cada nodo no tiene por qué tener asignada una única dirección IP, si no que puede ser responsable de una subred completa, dando acceso a la VPN a esa misma red, aunque este tipo de configuración no se tratará en este laboratorio.

La autoconfiguración de esta VPN se realiza intercambiando claves públicas entre los clientes y servidores que forman la red y usando un canal de comunicación para intercambiar metadatos sobre la red. Usando el cifrado asimétrico, TINC intercambia claves de cifrado simétricos para usarlo en la comunicación de datos entre los nodos. La figura 5 muestra un ejemplo de funcionamiento de una VPN con TINC, la conexión de los clientes a la red se inicia contra los dos nodos maestros marcados con una M. Inicialmente los clientes abren un canal de comunicación para intercambiar metadatos sobre la red (indicados en rojo). Esta comunicación de metadatos incluye la información necesaria para que los nodos puedan establecer una comunicación directa entre sí. Esta comunicación directa, están indicada en verde en la figura y corresponde a intercambio de datos entre nodos. En caso de estar dos nodos tras algún tipo de *firewall* y no poder comunicarse directamente, los nodos maestros intentan hacer de puente (relay) entre dichos nodos, este tipo de comunicación está indicada en naranja en la figura.

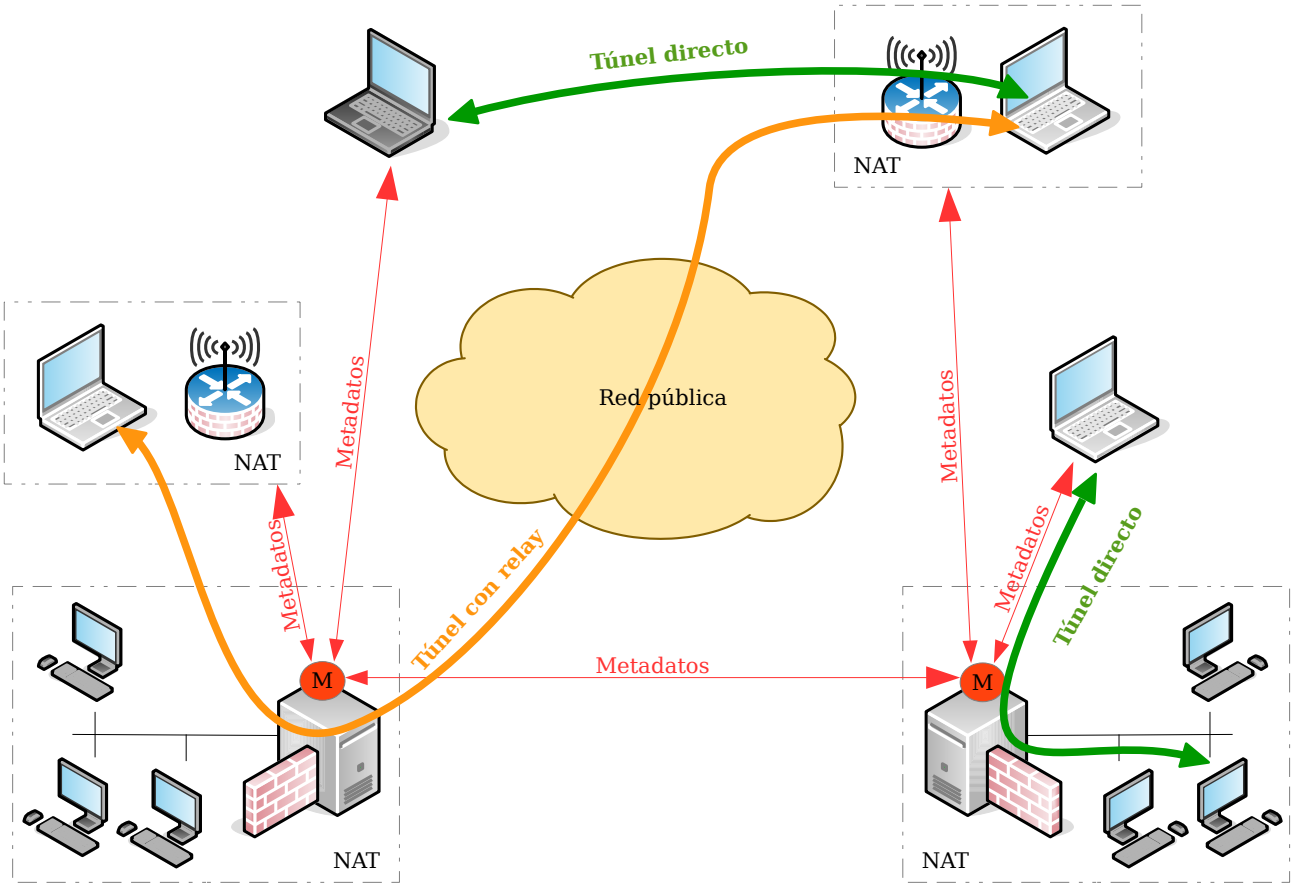


Figura 5. Ejemplo de comunicaciones en una VPN mesh con TINC.

En esta parte del laboratorio cada alumno no creará su propia VPN como se hizo con OpenVPN. Se propone conectarse a una única VPN creada por el profesor en la que los alumnos añadirán sus máquinas virtuales como nodos. Una vez conectados a la VPN de la asignatura, se proponen una serie de tareas para verificar el funcionamiento de VPN. Siga los siguientes pasos indicados a continuación para realizar la configuración de TINC:

Tarea 14.- Agregue reglas al *firewall* de la máquina GW que permita a los equipos internos conectar a equipos exteriores al puerto 655 usando el protocolo TCP. Compruebe el correcto funcionamiento de esta regla usando el comando `nc` conectando a un equipo externo en dicho puerto, la máquina compartida tiene TINC instalado y escucha en ese puerto.

Tarea 15.- Se procederá a agregar la máquina VM1 a la VPN creada por el profesor y llamada *redtai*. Para conectar los equipos restantes deberá seguir el mismo procedimiento:

T15.1.- Instale el paquete *tinc* en la máquina VM1

T15.2.- La conexión o creación de VPNs en TINC se consigue dando nombres a las redes. Cada nombre de red corresponde a un directorio dentro de */etc/tinc*. Para crear la nueva configuración para conectarse a *redtai* basta con crear el directorio */etc/tinc/redtai* con el comando `mkdir`.

T15.3.- Una vez creado el directorio debe crear un archivo de configuración dentro de éste mediante `nano /etc/tinc/redtai/tinc.conf` cuyo contenido debe ser el siguiente:

```
Name = uvusvm1
AddressFamily = ipv4
ConnectTo = profesor
```

Código 4. Configuración inicial de conexión a redtai con TINC.

T15.4.- En la primera línea del código anterior sustituya *uvusvm1* por uvus seguido del nombre de máquina (*vm1*) este nombre debe ser único para toda la VPN. **Importante**: Este nombre no puede contener caracteres especiales ni guiones por eso el uvus y el nombre de máquina deben ir sin espacios.

La última línea del fichero anterior contiene la directiva *ConnectTo*. Esta directiva indica un nodo maestro con el que intercambiar metadatos sobre la red distribuida. Inicialmente sólo está disponible el nodo del profesor, pero es posible añadir a los compañeros para mantener la VPN operativa en caso de desconexión del nodo inicial, para ello, deberá añadir líneas adicionales con los nodos e intercambiar los ficheros y se abordará al final de este laboratorio. Se procederá inicialmente a conectar a la VPN usando únicamente el nodo del profesor.

Tarea 16.- Para conectar al nodo maestro *profesor* es necesario conocer su dirección IP pública y su clave pública. Toda esta información deberá estar en un fichero llamado igual que el nodo al que se conectará, en este caso *profesor*.

T16.1.- Cree un directorio nuevo para almacenar los ficheros de los nodos maestros con `mkdir /etc/tinc/redtai/hosts` y cree un nuevo fichero dentro llamado *profesor*. Copie lo siguiente en dicho fichero:

```
Address = 192.168.20.3
Subnet = 172.16.0.0/22
Port = 655

-----BEGIN RSA PUBLIC KEY-----
MIICCgKCAgEAtigZmhAolvgMvR0s+MVPPodXIffhFkEtGCrQdRfvYHrDZSA6CMgW
0wc0HRekR0bWeRfKORGNdQgD/TS3iCGsf699ztVdnwLxeYVRM9a7qlU9MLUtBMTM
zdvs7vGsrNJkgFNXh3OY/qXH/ExqXrkiwok8pyj2WS9H2yVJnlJ9LVh7PmUcWZ2c
RqCKPqqh8iDp6L7HwIfz/CSA2w93/hlJSXu7lBvtGSNsafCp/XnKH51YpBKig1Et
sU+Q+yvkTNxKZ0pAMEi8aeCzFrWhbS9ygUtL8hIG2BYr/pWVgIz1oiJf5yq1pT9O
VLLyORMNcptl/KW6ipspi5Og7fme1S2RJFhGOzYbKSV0tCCodvsHy6bEONqAYrJh
+IbgetAeEkGgEE+hXFg6lMdFhvB6kfN02wI7m6G+xhO+ZmI4Ouj4oA5MkIGZfgE+
HfSlDueNdHMOOMFeFUvI8lpB59u9prQRnOUriJ6Iaa3/QDEQDejQ6niOfhR5d9L3
KoyCWe4QKzGeRYfPyjY8au2JfUXdWmO036oDMXnEi/vc2JxKjN51QBcMsCKWy7Yx
Ig7gO6OZA8iVhQv1czJfAdfRdHz58ymL1YBnhE0/NIh/UEJAd6Edv+ySgCmHbDWk
Tcl5X6GT1ks2eCCieFIjIw8GiGHGhO747IV2wqoOZtXpf0HwyYWwadsCAwEAAQ==
-----END RSA PUBLIC KEY-----
```

Código 5. Fichero profesor para la conexión con el nodo inicial.

T16.2.- Ahora el siguiente paso es crear una configuración para nuestro nodo cliente y un fichero con la clave pública, similar al mostrado paso anterior.

T16.3.- Cree un fichero nuevo en */etc/tinc/redtai/hosts* exactamente con el nombre indicado en la directiva *name* del código 4. Este fichero debe contener únicamente la dirección IP de la VPN que usará el equipo. Es importante no generar conflictos de direcciones IP, debe usar las 2 direcciones IP que

tenga asignadas para TINC, en la página de recursos de la asignatura. Con esta información, cree el fichero con el siguiente contenido (sustituyendo X por la IP asignada):

```
Subnet = 172.16.X.Y/32
```

Código 6. Fichero local con la dirección IP estática asignada al nodo cliente.

T16.4.- El siguiente paso es generar y anexar a esta configuración la clave pública. Esto se puede realizar automáticamente con el comando: `tincd -n redtai -K4096`. ¿En que fichero se ha generado la clave privada?, edite dicho fichero para ver su contenido.

T16.5.- Vuelva a abrir el fichero creado en el paso T16.3.- y compruebe si se ha anexado una clave pública.

T16.6.- Usando la plataforma de la asignatura suba el fichero al servidor TINC. Encontrará un formulario para este propósito en página de información de este laboratorio.

Llegado a este punto la red VPN no está operativa, queda indicar como configurar el interfaz de túnel que se creará, para ello, TINC ejecuta automáticamente el *script* llamado exactamente *tinc-up*, cuando se conecta a la VPN. En este *script* se debe configurar el adaptador de red con la IP asignada. Además puede añadir en este *script* rutas u otra configuración adicional a la red, pudiéndose alcanzar así redes adicionales a través de la VPN. Del mismo modo, en la desconexión de la VPN, TINC ejecuta el *script tinc-down*, en el cual debe desconfigurar la interfaz y eliminar cualquier ruta establecida previamente. Estos dos *scripts* deben estar en la ubicación correcta, por ello, sea cuidadoso al crearlos siguiendo los pasos indicados en las siguientes tareas.

Tarea 17.- Entre en el directorio */etc/tinc/redtai* y cree dos ficheros: *tinc-up* y *tinc-down* con el contenido mostrado a continuación:

```
#!/bin/sh
ip link set $INTERFACE up
ip addr add 172.16.X.Y/22 dev $INTERFACE
```

Código 7. Script tinc-up para configuración del adaptador túnel de la VPN.

```
#!/bin/sh
ip addr del 172.16.X.Y/22 dev $INTERFACE
ip link set $INTERFACE down
```

Código 8. Script tinc-down para desconfigurar la VPN.

T17.1.- Estos *scripts* necesitan tener permiso de ejecución (x), para ser lanzados por TINC. Debe usar el comando `chmod +x tinc-up tinc-down` para establecer los permisos.

T17.2.- Para comprobar si se ha configurado correctamente se ejecutará TINC en modo de depuración en un terminal. En un nuevo terminal ejecute el comando `tincd -n redtai -D -d1`. Considere que este

terminal se quedará bloqueado por TINC y no se puede parar ni usando CONTROL+C.

T17.3.- En otro terminal compruebe si aparece un nuevo adaptador llamado redtai con la IP configurada anteriormente (comando `ip a`). Si no aparece debe revisar el fichero *tinc-up*.

T17.4.- De nuevo en otro terminal use el comando *ping* hacia 172.16.0.1 que es el equipo del profesor y así comprobar si está conectado a la VPN. Si no funciona debe revisar los mensajes de depuración del terminal donde está ejecutándose TINC.

T17.5.- Para parar TINC cuando está modo de depuración, debe ejecutar el comando `pkill tincd` desde otro terminal.

T17.6.- El siguiente paso es iniciar la red VPN como servicio del sistema, ejecute el comando `systemctl start tinc@redtai` y compruebe con `ip a` si de nuevo aparece una interfaz llamada *redtai* con la IP correspondiente.

T17.7.- Para automatizar el inicio de la VPN con el inicio del sistema debe ejecutar `systemctl enable tinc@redtai`. Compruebe si funciona reiniciando la máquina.

Una vez conectada una máquina a la VPN debe repetir le proceso para añadir la restante. Además, se propone como ejercicio hacer un escaneo de la red para localizar a los compañeros que ya están en la VPN y buscar un equipo oculto que contiene un Wiki.

Tarea 18.- Añada la máquina VM2 y VM3 a la VPN asegurándose que se conectan automáticamente tras el reinicio.

Tarea 19.- Opcional-4d: Intente localizar un equipo en la VPN que contiene un Wiki, una vez que lo localice, navegue el Wiki y deje una reseña indicando su nombre y como lo ha encontrado. Use el ejemplo existente en la propia Wiki y no borre las reseñas de sus compañeros.

4.1. Instalación de un nodo maestro (opcional)

En esta sección se muestra como instalar un nodo maestro que permita salvaguardar la red cuando el nodo del profesor no esté disponible. Este nodo maestro debe ejecutarse en un equipo que tenga una dirección IP pública conocida, por ello, se propone redirigir adecuadamente los puertos a VM2 para que esta máquina sea un nodo maestro.

Tarea 20.- Opcional-4e: Esiga los siguientes pasos para convertir el nodo TINC de la máquina VM2 en un nodo maestro usando su IP pública.

T20.1.- Redirija el puerto TCP de TINC mediante DNAT desde la IP pública de GW a la máquina VM2.

T20.2.- En la máquina VM2 edite el fichero */etc/tinc/redtai/hosts/uvusvm2* y añada la directiva *Address* con su dirección IP pública, fíjese como se ha hecho en el código 5.

T20.3.- Este fichero debe intercambiarlo con algún compañero que haya realizado el mismo procedimiento. Puede utilizar la propia Wiki existente en la VPN para pegar el contenido de estos ficheros y poder intercambiarlos. Suba el contenido de este fichero al Wiki de la VPN.

T20.4.- Use el fichero TINC con la clave pública de algún otro compañero y cópielo a

/etc/tinc/redtai/hosts. A partir de este momento su compañero puede conectarse para usar su equipo como nodo maestro.

T20.5.- Una vez intercambiadas las claves públicas edite la configuración de TINC en */etc/tinc/redtai/tinc.conf* y añada tantas directivas *ConnectTo* como compañeros con los que intercambie las claves públicas. Fíjese en el ejemplo:

```
Name = wiki
AddressFamily = ipv4
ConnectTo = profesor
ConnectTo = auregw
ConnectTo = fomargw
ConnectTo = takigw
```

Código 9. Fichero tinc.conf con conexión a múltiples nodos.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 5 - QoS y Control de tráfico

Enunciados de laboratorios
Tecnologías Avanzadas de la Información
Rev. 1 - 29/11/23

1. Introducción y objetivos

El tiempo estimado de realización de este laboratorio es de **4 horas** y está centrado en presentar sistemas de control de tráfico relacionados con QoS en Linux. Linux implementa en su núcleo mecanismos para gestionar el tráfico con multitud de posibilidades. Algunas de las implementaciones incluidas no están ampliamente probadas, y de otras se conocen algunas deficiencias en su funcionamiento. A pesar de ello, conociendo las limitaciones existentes se pueden conseguir resultados satisfactorios tras realizar pruebas con diferentes configuraciones. Ante la multitud de posibilidades existentes, en este laboratorio se describirán sólo algunas probando un conjunto reducido de ellas, concretamente las más relacionadas con la parte teórica de la asignatura.

La parte práctica de este laboratorio se ha dividido en dos grandes bloques, el primero está completamente guiado para facilitar la comprensión de la configuración y de los resultados obtenidos al aplicar diferentes disciplinas. En un segundo bloque, se requiere realizar la configuración de una disciplina descrita esquemáticamente donde se deben aplicar los conocimientos adquiridos. Como en el resto de los laboratorios, para facilitar el desarrollo de la sesión se debe disponer del material de la tabla 1 en la ubicación indicada.

	Descripción	**Ubicación**
descargas.sh, descargas2.sh	Script para realizar transferencias simultáneas	Equipo local o Máquina compartida
escucha-puerto.sh	Script para emular un servicio en un puerto TCP	GW

Tabla 1. Material necesario para la realización de la sesión de laboratorio.

2. Implementación en Linux del control de tráfico

La documentación de referencia para técnicas avanzadas de red en Linux es LARTC (Linux Advanced Routing & Traffic Control) disponible en https://www.lartc.org/. Esta documentación es extensa, algunas partes están incompletas y se recomienda utilizar la versión en inglés, ya que las traducciones no son de buena calidad. La mayor parte del contenido aquí expuesto se encuentra a lo largo del capítulo 9 de esta guía, pero se añaden en esta documentación algunas consideraciones fruto de la experiencia con el uso de estas herramientas de Linux que no se tratan en la documentación oficial.

La implementación del control de tráfico en Linux presenta cierta complejidad, pues está pensada para facilitar a los desarrolladores la implementación de nuevas disciplinas para el control del tráfico. De hecho, aparecen en el núcleo multitud de tipos de colas y disciplinas con funcionalidad diferente y desarrolladas por diferentes personas, e incluso, algunas experimentales no ampliamente probadas. A pesar de la diversidad, todas las implementaciones son homogéneas respecto al modo de configuración, realizándose todas con la misma herramienta (*tc*) facilitando al administrador del sistema su configuración. Esta herramienta está ampliamente documentada en las páginas de manual de Linux.

Básicamente, el control de tráfico en Linux realiza cuatro operaciones: conformado de tráfico, reordenación de paquetes, vigilancia y eliminación de paquetes. Estas operaciones se realizan configurando cada interfaz de red con múltiples disciplinas, cada disciplina puede llevar asociada una o varias colas, y los paquetes que llegan a la interfaz se ubican en alguna de las colas existentes. La cola inicial para cada paquete depende de la clasificación. El núcleo decide cual es la disciplina y la cola inicial para cada paquete mediante listas de reglas NFT o *tc*, estos últimos llamados filtros.

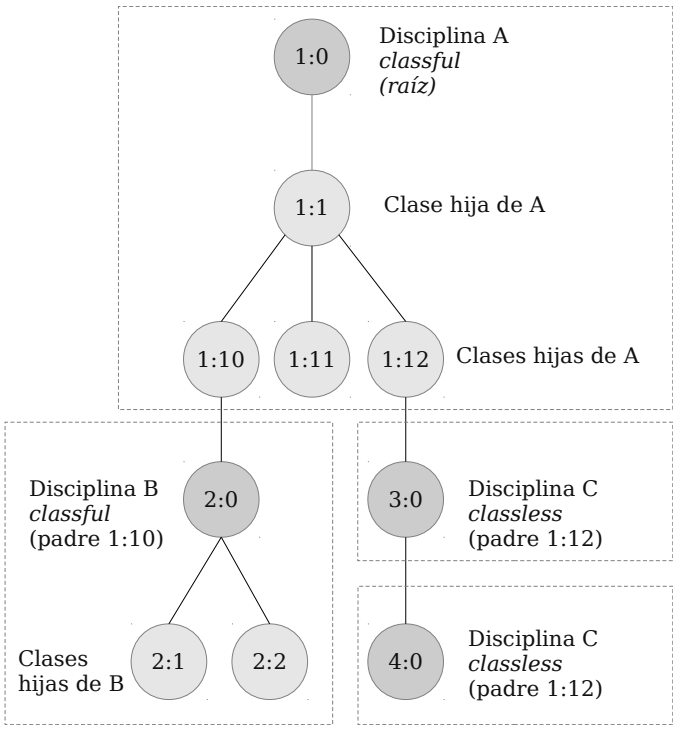


Figura 1. Jerarquía de ejemplo

En la implementación realizada en Linux las disciplinas de colas que se deseen usar se asocian en forma de

árbol, formando una jerarquía de disciplinas cuyo nodo raíz es una interfaz de red. Si se utiliza la herramienta *tc*, a ciertos nodos se adjuntan reglas para decidir cual es el nodo destino de cada paquete, estas reglas se denominan filtros, pero en este laboratorio se realizará la clasificación usando reglas NFT. Algunas disciplinas de colas van más allá de ser un simple nodo de la jerarquía, incluyen subnodos llamadas clases hijas donde se pueden encolar los paquetes.

En la figura 1 se muestra esquemáticamente una posible configuración con diferentes disciplinas y clases. La clave para entender este mecanismo es profundizar en los tres tipos de elementos que forman parte del árbol: disciplinas y clases.

Las clases siempre forman parte de una disciplina, no pueden existir fuera de ellas. Es fácil de entender con el siguiente ejemplo: suponga que se desea utilizar una *disciplina de colas de prioridad* con 5 colas de prioridad diferentes; en vez de crear 5 disciplinas en el árbol, la disciplina *cola de prioridad* incluye la posibilidad de añadir clases hijas donde, cada clase hija contiene una cola diferente con una prioridad asignada. Así, con sólo una disciplina se consigue esta configuración.

Pero no todas las disciplinas implementadas disponen de clases hijas, distinguiéndose dos tipos: disciplinas sin clasificación (*classless*) y disciplinas con clasificación (*classful*). Para no crear confusión entre *classless* y *classful* se define una disciplina *classless* como aquella que no puede subdividirse en clases hijas. Para las disciplinas que contienen clases, estas clases aparecen en el árbol de jerarquía como nodos hijos de las disciplinas quedando el árbol formado sólo por dos tipos de nodos, las disciplinas y las cases hijas de las disciplinas. Es importante resaltar que todas las disciplinas de colas en tienen funcionalidades simples y son: aceptar los datos, reordenar su envío, retrasarlo o eliminarlo de la cola. Como ya se indicó, toda esta funcionalidad se manipula con el comando *tc*, y por cada disciplina disponible existe una página de manual adicional donde se precisa la configuración de la misma y de las posibles clases hijas.

Otro aspecto importante es conocer como el sistema de clasificación identifica los nodos inequívocamente para poder clasificar cada paquete en un nodo. La solución adoptada consiste en usar un identificador único en cada elemento de la jerarquía, este identificador único es una pareja de números llamados número mayor y número menor, separados por “:”. Éstos serán asignados explícita o implícitamente durante la configuración. Los números mayores siempre identifican las disciplinas, así, cada disciplina tendrá un número mayor diferente. En cambio, el número menor identifica a cada clase hija de una disciplina, teniendo siempre cada clase hija el número mayor de la disciplina a la que pertenece.

Una vez vistos los aspectos generales las disciplinas de colas, se enumeran las disciplinas sin clasificación (*classless*) más relevantes disponibles:

- PFIFO y BFIFO: Disciplina simple basada en una cola FIFO de un tamaño fijo en paquetes (PFIFO) o bytes (BFIFO).
- PFIFO_FAST: Disciplina estándar consistente en una cola FIFO dividida en tres bandas de prioridad relacionadas con el campo TOS de los paquetes IPv4.
- RED: *Random Early Detection*, disciplina para simular la congestión de un enlace eliminando paquetes a azar.
- SFQ: *Stochastic Fairness Queueing*, es una disciplina llamada estocástica equitativa basada en la auto-detección de flujos, reordena los paquetes para tratar equitativamente a cada flujo.

- TBF: *Token Bucket Filter*, disciplina de cubeta con fichas para regulación de velocidad.
- CHOKE: *CHOose and Keep for responsive flows*, disciplina para usar en situaciones de congestión, identifica y penaliza los flujos que monopolizan el enlace.

Las disciplinas de colas con clasificación (*classful*) más relevantes son:

- PRIO: Disciplina de cola con bandas de prioridad configurables, no implementa regulación de velocidad.
- CBQ: *Class Based Queueing*, es una disciplina con clases hijas para compartir el ancho de banda, compleja y con multitud de parámetros como son: prioridad, límites de ancho de banda, etc.
- HTB: *Hierarchy Token Bucket*, esta disciplina también sirve para compartir el ancho de banda entre clases hijas garantizando ancho de banda en cada clase además de priorizar y regular velocidad basado en TBF.
- DRR: *Deficit Round Robin Scheduler*, es una disciplina con mayor flexibilidad pensada para reemplazar a SFQ, la diferencia es que DDR no contiene de manera predeterminada ninguna cola. Básicamente su funcionamiento consiste en descartar cualquier paquete que no esté clasificado.

Además de las enumeradas anteriormente existen algunas otras que se pueden consultar en la documentación de Linux y en la página de manual del comando *tc*.

En las siguientes secciones se mostrarán los aspectos más relevantes de las diferentes disciplinas pero entrando sólo a detallar aquellas utilizadas en la sección práctica. Para poder entender los ejemplos de configuración, en primer lugar se describe brevemente la herramienta *tc* que se debe utilizar como administrador para la configuración.

2.1. Herramienta TC

Toda la configuración del control de tráfico se realiza con el comando *tc*. El uso general de este comando tiene tres modos y cada uno permite manipular un elemento diferente del sistema de control de tráfico: disciplinas, clases y filtros (aunque este último no será necesario). La sintaxis para configurar disciplinas y clases depende de cada disciplina usada, y se mostrarán diversos ejemplos en la descripción de cada disciplina considerada de interés.

De forma general, se utilizará el comando *tc* según lo indicado en la tabla 2, y se debe considerar un aspecto adicional que son las unidades utilizadas con este comando, ya que pueden crear confusión. En la tabla 3 se muestra el significado de las unidades, las cuales, se puede consultar en la página principal de manual de *tc*.

Modo	Comando (ejemplo)	Descripción
Disciplina	**tc qdisc add dev** eth0 **root** tbf ...	Añadir disciplina TBF como raíz
	tc qdisc add dev eth0 **parent** 1:0 **handle** 10: sqf ...	Añadir disciplina SFQ como hija de la disciplina 1:0
	tc qdisc del dev eth0 **root**	Eliminar todas las disciplinas de la interfaz eth0 (limpieza)

Modo	Comando (ejemplo)	Descripción
Clase	**tc class add dev** eth0 **parent** 1:0 **classid** 1:1 **htb** ...	Añadir clase hija a la disciplina 1:0
Filtro	**tc filter add dev** eth0 **parent** 1:0 **protocol** ip …	Añade un filtro para clasificación a la disciplina 1:0
Estadísticas	**tc** -s -d **qdisc show dev** eth0	Muestras las disciplinas de la interfaz eth0
	tc -s -d **class show dev** eth0	Muestra las clases de la interfaz eth0
	tc -s -d **filter show dev** eth0	Lista los filtros de la interfaz eth0

Tabla 2. Modos de operación del comando tc.

Sintaxis	Unidad	Equivalencia
bps	Bytes por segundo / Bytes	
kbps	Kilobytes por segundo / Kilobytes	1000bps
mbps	Megabytes por segundo / Megabytes	106bps
gbps	Gigabytes por segundo / Gigabytes	109bps
bit	Bits por segundo	
kbit	Kilobits por segundo / Kilobits	1000bits
gbit	Gigabits por segundo / Gigabits	106bits
mb	Megabytes	109bits
s, sec, secs	Segundos	
ms, msec, msecs	Milisegundos	10-3 sec
us, usec, usecs	Microsegundos	10-6 sec

Tabla 3. Unidades utilizadas en el comando tc.

2.2. Disciplinas sin clasificación

Las disciplinas sin clasificación no disponen de la posibilidad añadir clases hijas para clasificar paquetes. Estas disciplinas tienen dos usos generales: en la raíz de la interfaz y en las hojas del árbol de clasificación.

Estas disciplinas se utilizan conjuntamente con disciplinas que sí admiten clasificación, de forma que una disciplina con clasificación y múltiples clases hijas admite una disciplina sin clasificación debajo de sus clases de último nivel. Se mostrará la utilidad de este tipo de configuración en la sección práctica.

2.2.1. Disciplina PFIFO / BFIFO

Las disciplinas más básicas existentes son PFIFO y BFIFO y son simples colas FIFO cuya política de envío es *Fisrt In – Fisrt Out*. La primera letra P o B hace referencia a paquetes o bytes respectivamente. En

su configuración sólo admite como parámetro el tamaño de la cola (parámetro *limit*), y a pesar de su simpleza, su uso es recomendado en múltiples situaciones. Otra característica importante es que mantiene estadísticas del funcionamiento de la cola y pueden consultarse fácilmente. Una disciplina de este tipo puede ser añadida mediante el comando:

```
tc qdisc add dev eth0 root pfifo limit 10
```

2.2.2. Disciplina PFIFO_FAST

Es la disciplina utilizada en Linux de forma predeterminada en cada interfaz, Como en el caso anterior, la política de envío es *Fisrt In – Fisrt Out* pero, con una modificación consistente en la división en 3 partes de la cola, llamadas bandas. No se estudiará en profundidad ya que no se usará en esta sesión de laboratorio, pero se resumen algunos detalles sobre su modo de operación y se puede consultar una amplia documentación en la página de manual *tc-pfifo_fast*.

El funcionamiento básico de esta disciplina es la siguiente: el núcleo minimiza el retraso para los paquetes de la banda 0, por ello, mientras hay paquetes en la banda 0, las bandas 1 y 2 no son procesadas, y para procesar la banda 2 deben estar vacías las bandas 0 y 1. Los paquetes son introducidos automáticamente en las bandas según una configuración establecida por el administrador (parámetro *priomap*) consistente en mapear cada uno de los posibles valores del campo TOS del paquete a cada banda, al existir 4 bits TOS aparecen 16 valores para mapear.

Esta disciplina se confunde a veces con una disciplina *classful*, pero no lo es. En PFIFO_FAST no se puede realizar clasificación manual de paquetes, la clasificación es automática, se asignan paquetes a cada banda analizando únicamente el campo TOS, no es posible clasificarlos de ningún otro modo.

En la documentación de esta disciplina se detalla cual es el mapa de prioridad establecido de manera predeterminada. Este mapa está relacionado con el significado de los cuatro bits TOS: minimizar retraso, maximizar caudal, maximizar fiabilidad y minimizar coste.

2.2.3. Disciplina cubeta con fichas TBF

Esta disciplina ha sido ampliamente estudiada en la parte teórica de la asignatura y es implementada en Linux con algunas peculiaridades. La disciplina TBF (*Token Bucket Filter*) controla el caudal de salida con gran precisión y permite pequeñas ráfagas bajo ciertas condiciones.

Para hacer un uso adecuado de esta disciplina se debe establecer correctamente la velocidad de entrada de fichas y el tamaño de la cubeta, siendo consciente del efecto que tienen estos parámetros:

- La velocidad de entrada de fichas regula con precisión la velocidad de salida.
- El tamaño de la cubeta indica la cantidad máxima de paquetes que pueden salir en una ráfaga.

Con la implementación realizada en Linux se contempla una correspondencia entre fichas y bytes en relación uno a uno, es decir, una ficha es un byte. Además de los parámetros básicos como tamaño de cubeta, tamaño de cola y fichas por segundo, esta implementación contempla parámetros adicionales con el objetivo de

aumentar la precisión sobre el control del flujo de salida de la cubeta. Es fundamental utilizarlos adecuadamente y comprender el efecto causado en el flujo de salida la alteración de cada uno de ellos. Son los siguientes:

- Tamaño de cola o latencia (*limit / latency*): establece el tamaño de la cola, es posible en la configuración indicar la latencia en vez del tamaño. Dada una latencia, el sistema calcula el tamaño de cola automáticamente.
- Tamaño máximo de ráfaga (*burst*): corresponde al tamaño de la cubeta en bytes. Si se establece este parámetro muy pequeño se perderán muchos paquetes que no cabrán en la cola si entra una ráfaga, y no se regulará la velocidad correctamente. Según la documentación se debe considerar una relación mínima en enlaces de 10Mbits/s al menos 10KBytes.
- Tamaño mínimo del token (*mpu*): este parámetro se utiliza para considerar que los paquetes sin datos sí consumen ancho de banda, por ejemplo, el tamaño mínimo de la trama ethernet, incluyendo únicamente cabeceras, es de 72 bytes. Debe establecerse al valor en bytes consumido por un paquete vacío.
- Velocidad o tasa (*rate*): Número de bytes o bits (según unidad usada) por segundo con el que se llena la cubeta.
- Velocidad pico (*peakrate*): Este parámetro tiene sentido cuando la cubeta contiene fichas y se emite una ráfaga. Se podrían quitar todas las fichas y los paquetes en la mínima unidad de tiempo de computación y se obtendría una velocidad de ráfaga prácticamente infinita, produciendo efectos indeseados. Este parámetro habitualmente no es necesario establecerlo al ser automáticamente calculado a partir de los otros.
- Cubeta de ráfaga (*mtu/minburst*): El parámetro anterior (*peakrate*) limita la velocidad de la ráfaga, así surge una cubeta secundaria encargada de limitar la velocidad de ráfaga. Se debe establecer al tamaño MTU de la interfaz de red para un comportamiento óptimo.

En la mayoría de los casos no es necesario afinar la configuración estableciendo todos los parámetros descritos. De hecho, muchos de ellos son calculados automáticamente, así que para una configuración típica en una disciplina TBF, basta con especificar los indicados en el siguiente ejemplo:

```
tc qdisc add dev eth0 root tbf rate 630kbit latency 50ms burst 1540
```

2.2.4. Cola estocástica equitativa SFQ

Este tipo de disciplina es una implementación de la disciplina CFQ estudiada en la parte teórica de la asignatura, presenta algunas diferencias en la implementación. Con CFQ se establecían diferentes colas de envío equitativas, donde un sistema de clasificación se encarga de añadir los paquetes en cada cola. Usando un mecanismo *round robin* con un *quantum* determinado se extraían paquetes de cada cola de forma circular. La diferencia es que con SFQ las colas son creadas automáticamente y el tráfico también es clasificado automáticamente. Esta disciplina es muy recomendable en enlaces de salida congestionados ya que equilibra todos los flujos.

La implementación realizada en Linux realiza una auto-clasificación de flujos, el comportamiento es equivalente a crear de forma dinámica una cola diferente para cada cada flujo (sesión) existente, ya sea TCP o UDP, y a las colas se les aplica el algoritmo CFQ. Con esta implementación se obtiene un tratamiento equitativo para cada flujo. En la implementación se simplifican la colas, no creando en realidad una cola para cada flujo, se utilizan funciones *hash* para identificar y clasificar paquetes de cada flujo, todos los paquetes residen en una estructura de datos optimizada para su acceso con las funciones *hash*.

El uso de funciones *hash* para selección de paquetes rompe la equidad en intervalos de tiempo grandes, por ello, esta implementación cambia cada cierto tiempo la función *hash* utilizada, este tiempo de cambio de función es considerado un parámetro configurable en este tipo de disciplina.

La configuración de una disciplina SFQ es simple y sólo hay que contemplar estos parámetros:

- Tiempo de cambio de la función hash (*perturb*): De forma predeterminada se establece a 10 segundos, no se recomienda cambiarla aunque los valores en el intervalo 5-10 son razonables.
- Quantum: Cantidad de bytes extraídos de cada flujo en el turno, es importante no establecer este parámetro por debajo de la MTU, podría quedarse un flujo sin recibir servicio.
- Tamaño de cola (*limit*): Cantidad de paquetes totales que pueden almacenarse en la cola.
- Divisor: Permite establecer el tamaño de tabla *hash*, no se recomienda su utilización.

Como en casos anteriores se pueden omitir parámetros en la configuración y dejar que el sistema los establezca adecuadamente, un ejemplo sería el siguiente:

```
tc qdisc add dev eth0 root sfq perturb 10
```

2.3. Disciplinas de colas con clasificación

Estas disciplinas son más complejas y permiten ser el destino de la clasificación de paquetes a lo largo del árbol de clases hijas. A su vez, las clases hijas admiten como hijos otras disciplinas adicionales siendo esta configuración, aunque parezca compleja, de uso muy común ya que aporta beneficios en el comportamiento de los flujos.

Se puede generalizar el comportamiento de cada clase como un conformador y/o planificador de tráfico: regula la velocidad de salida y establece un orden de salida de paquetes. Existe una disciplina llamada PRIO cuya finalidad no es la regulación, sólo la reordenación en la salida de paquetes y otras como CBQ y HTB que implementan tanto la regulación como la planificación.

Las principales disciplinas clasificadoras de este tipo son las tres mencionadas: PRIO, CBQ y HTB pero se estudiará en mayor profundidad HTB, por un lado porque presenta mejoras significativas respecto a CQB, tanto a nivel de rendimiento como a nivel de configuración, y por otro lado, porque existe configuraciones de HTB equivalentes a PRIO.

2.3.1. Disciplina PRIO

El comportamiento de esta disciplina es similar a PFIFO_FAST pero se pueden especificar el número de bandas deseadas. Por cada banda, automáticamente se crea una clase donde poder clasificar los paquetes. Cuando se configura la disciplina se puede indicar el número de bandas y el mapa de prioridad, aunque si no se especifican las bandas, se establecen tres y un mapa predeterminado. Así, los parámetros de configuración son *bands* y *priomap* respectivamente.

En el siguiente ejemplo se crean 4 bandas de prioridad estableciendo como número mayor de esta disciplina el número 1:

```
tc qdisc add dev eth0 root handle 1: prio bands 4
```

La ejecución de este comando crea la estructura mostrada en la figura 2 con 4 bandas de prioridad. Los números menores de las clases hijas son asignados automáticamente desde 1 a 4. Recuerde que el comportamiento es como el de PFIFO_FAST, las bandas de menor número son las más prioritarias y el modo de operación consiste en atender los paquetes de cualquier banda, si y sólo si, no hay paquetes en las bandas más prioritarias (aquellas con inferior número menor).

Se puede consultar la configuración establecida de clases por el sistema solicitando información con la herramienta *tc* sobre las disciplinas y clases existentes. Considere el ejemplo de la salida mostrada por los dos siguientes comandos y la correspondencia con la figura 2:

```
# tc -d qdisc show dev eth0
qdisc prio 1: root refcnt 2 bands 5 priomap  1 2 2 2 1 2 0 0 1 1
1 1 1 1 1 1

# tc -d class show dev eth0
class prio 1:1 parent 1:
class prio 1:2 parent 1:
class prio 1:3 parent 1:
class prio 1:4 parent 1:
class prio 1:5 parent 1:
```

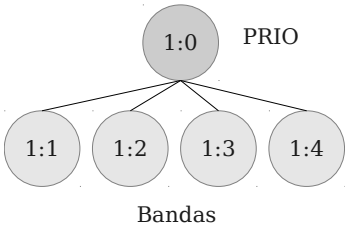


Figura 2. Disciplina PRIO con 4 bandas

2.3.2. Disciplina CBQ

CBQ fue uno de los primeros clasificadores implementados en Linux pero ampliamente criticado tanto por desarrolladores como administradores de sistemas. Desde el punto de vista del administrador su utilización y configuración es extremadamente compleja, y desde el punto de vista de implementación se comprueba que no aprovecha en su totalidad el ancho de banda disponible y sobrecarga en exceso el sistema.

Básicamente CBQ consigue la regulación de velocidad dejando de emitir paquetes durante ciertos intervalos de tiempo, se dice que deja en reposo su salida. Las configuraciones realizadas con CBQ se pueden realizar de manera equivalente con HTB. HTB presenta dos ventajas principales frente a CBQ, la primera es una mayor facilidad de configuración, y la segunda, es una mayor eficiencia en el uso de CPU al realizar los cálculos de manera simplificada respecto a CBQ.

Por ello, no se detallará su uso y se centrará el laboratorio en el uso de HTB.

2.3.3. Disciplina HTB

Esta disciplina es óptima cuando se dispone de un ancho de banda fijo y estable, configurando HTB adecuadamente se puede dividir el caudal disponible en partes con una gran exactitud. El modo de operación de HTB consiste en garantizar un ancho de banda mínimo configurado para cada una de las subdivisiones, si hay excedente de ancho de banda, se cede a otras clases el ancho de banda sobrante pero en el reparto del sobrante, sólo se permite a cada clase alcanzar un máximo establecido en la configuración. Además de controlar el máximo/mínimo caudal en cada clase se implementa un sistema de prioridades en el reparto de ancho de banda sobrante consiguiéndose comportamientos equivalentes a la disciplina PRIO.

Básicamente el funcionamiento se puede presentar como una jerarquía de clases, donde las clases padres reparten ancho de banda a las clases hijas, siguiendo determinadas reglas. Para configurar esta disciplina se parte de un nodo raíz HTB que admite una única configuración que es la clase predeterminada donde se encola el tráfico. Tras esto, se configuran todas las clases hijas que se deseen en forma de árbol y en cada uno de los hijos del árbol se establecen los siguientes parámetros: caudal mínimo garantizado, caudal máximo alcanzable y prioridad en la adquisición de caudal disponible.

Para hacer un uso adecuado de HTB se requiere conocer el algoritmo seguido cuando se extrae un paquete para su envío desde alguna clase del árbol HTB. El primer principio de operación es conocer una de las restricciones de HTB: todos los paquetes residen siempre en colas ubicadas en las hojas del árbol de clases HTB, por tanto, en los nodos internos nunca existen colas de paquetes. Una hoja es un nodo del árbol HTB que no tiene hijos.

Con la restricción indicada, el algoritmo de operación en primer lugar reparte el caudal disponible en la jerarquía HTB siempre de padres a hijos y de dos formas: verticalmente entre clases padres e hijas y horizontalmente entre clases hermanas. Las reglas de distribución de caudal seguidas son las siguientes:

- Una clase hija siempre envía paquetes con el caudal mínimo garantizado.
- Una clase hija solicita ancho de banda al nodo padre para intentar llegar a su máximo caudal.
- Un padre suma los caudales que está recibiendo de sus hijos y si es menor que su caudal máximo reparte el sobrante entre los hijos.

En segundo lugar, el algoritmo usa un método de reparto del ancho de banda sobrante desde el padre a los hijos siguiendo un proceso basado en prioridades. Así, cada clase hija tiene configurada una prioridad, y en igualdad de prioridad se aplica *round robin* entre hijos con un *quantum* precalculado. Aunque el *quantum* puede ser configurado manualmente, no se recomienda.

Tras estudiar el modo de operación, el proceso de configuración también presenta algunas peculiaridades. Para realizarlo correctamente se indican las principales restricciones de una jerarquía HTB en la estructura del árbol de clases:

- Los nodos intermedios del árbol no pueden contener colas de paquetes, por ello, el destino de la clasificación no pueden ser estos nodos.

- En las hojas del árbol (nodos que no tienen hijos) se puede indicar la disciplina de cola a usar, aunque de forma predeterminada se establece una cola PFIFO.

Para entender mejor lo expuesto se analiza la configuración HTB mostrada en la figura 3. En este ejemplo se distribuye un ancho de banda de 10MB entre tres equipos A, B y C a los que se aplican diferentes políticas. Los parámetros utilizados por HTB son *rate, ceil* y *prio* cuyo significado es:

- Rate: Caudal garantizado.
- Ceil: Caudal máximo alcanzable.
- Prio: Prioridad en la obtención del caudal sobrante.

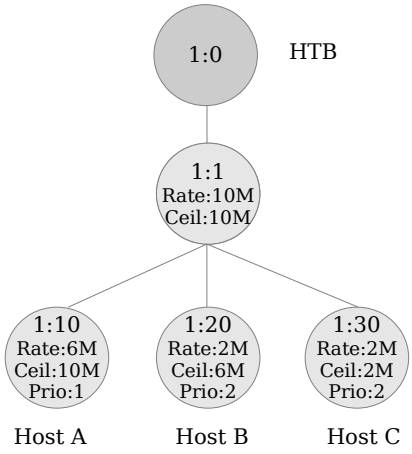


Figura 3. Jerarquía de ejemplo HTB

Con la configuración y parámetros mostrados en la figura 3 cada equipo tiene la siguiente política asignada:

- Host A:
  - Tiene garantizado un caudal de 6MB.
  - Puede llegar a utilizar un máximo caudal de 10MB si hay caudal disponible.
  - Este equipo tiene prioridad 1 en la solicitud de ancho de banda sobrante a la clase padre.
- Host B:
  - Tiene garantizado un caudal de 2MB.
  - Utilizará un máximo caudal de 6MB si hay caudal disponible en el nodo padre.
  - Este equipo tiene prioridad 2 en la solicitud de ancho de banda sobrante.
- Host C:
  - Tiene garantizado un caudal de 2MB.
  - Utilizará un máximo caudal de 2MB.
  - Nunca solicita caudal extra a la clase padre por tener su caudal máximo igual al garantizado.

Fíjese en la restricción establecida en la configuración del equipo C, tiene el mismo valor establecido para el caudal garantizado que para el caudal máximo, esto significa que no participará nunca en el proceso de solicitud de ancho de banda a la clase padre ya que nunca pedirá caudal extra a la clase padre.

Para ilustrar con mayor profundidad el ejemplo de la figura 3, a continuación se describe el comportamiento de esta configuración en algunas situaciones, considere que existen más posibilidades de comportamiento para los tres equipos a parte de los siguientes tres casos:

- Caso 1: Todos los equipos intentan utilizar el máximo ancho de banda: como *(Rate 1:10) + (Rate 1:20) + (Rate 1:20) = (Rate 1:1)* la clase 1:1 no dispone de ancho de banda adicional para distribuir entre los hijos. El resultado es que todos los equipos disponen sólo del ancho de banda garantizado.
- Caso 2: El equipo C está en reposo, A y B intentan utilizar el máximo ancho de banda posible: como

(*Rate 1:10) + (Rate 1:20) = 8MB* la padre clase 1:1 dispone de 2MB adicionales para repartir entre los hijos. La clase 1:1 reparte el caudal sobrante entre los hijos, pero por orden de prioridad, hasta que cada hijo alcance a su parámetro *Ceil*, por tanto, cede los 2MB a clase 1:10. El resultado es que el equipo A usa un caudal de 8MB y el equipo B sólo dispone de su caudal garantizado de 2MB.

- Caso 3: El equipo A está en reposo, C y B intentan disponer del máximo caudal posible. En esta situación *(Rate B) + (Rate C) = 4M*, así, la clase padre 1:1 dispone de 6MB adicionales para repartir entre hijos. El hijo más prioritario (A) no solicita caudal y los dos restantes, con igual prioridad solicitan hasta alcanzar su parámetro *Ceil*. El resultado en esta situación es que el equipo B usa un caudal de 6MB y el equipo C 2MB, ambos alcanzan el caudal *Ceil* (máximo) y a la clase padre le sobran 2MB que no utiliza.

Existen algunas situaciones en las que el comportamiento puede ser el no deseado, en el último caso sobra caudal debido a las restricciones impuestas a los equipos. Suponga ahora que la suma de los caudales garantizados (*rate*) de todos los hijos es mayor que el caudal garantizado del padre; esta situación es no deseable y debe evitarse, ya que pierde sentido el parámetro *rate* como caudal garantizado, al no poder la clase padre garantizar el caudal de los hijos. Para completar el ejemplo se muestra la secuencia de comandos *tc* para conseguir la configuración mostrada en la figura 3:

```
tc qdisc add dev eth0 root handle 1: htb default 30
tc class add dev eth0 parent 1: classid 1:1 htb rate 10mbit
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 6mbit ceil 10mbit prio 1
tc class add dev eth0 parent 1:1 classid 1:20 htb rate 2mbit ceil  6mbit prio 2
tc class add dev eth0 parent 1:1 classid 1:30 htb rate 2mbit ceil  2mbit prio 2
```

La existencia de la clase 1:1 es otra de las restricciones de HTB, se debe a que sólo se puede conformar tráfico en las clases hijas. En la especificación de la disciplina HTB sólo se puede indicar la clase por defecto pero no ningún otro parámetro relacionado con el conformado del tráfico. Con esto se concluye que siempre la disciplina HTB está obligada a tener al menos una clase hija donde se establezcan los parámetros de configuración *rate*, *ceil*, etc.

Usando adecuadamente esta disciplina existen equivalencias a las disciplinas TBF y PRIO presentadas con anterioridad, las equivalencias son las siguientes:

- Conseguir una disciplina TBF con HTB es equivalente a tener una clase HTB con el parámetro *rate* y *ceil* de igual valor, este ejemplo se mostró en el ejemplo de la figura 3.
- Conseguir una disciplina PRIO con HTB es equivalente a crear una clase hija HTB por cada banda de prioridad estableciendo los parámetros *prio* de cada clase hija adecuadamente.

Para finalizar se muestran algunos parámetros adicionales permitidos en la configuración de las clases HTB. Cuando se omiten parámetros durante la configuración, el sistema los establece a valores precalculados, pero en algunas situaciones es útil alterar su valor. Los parámetros indicados en la tabla 4 son los principales para esta disciplina.

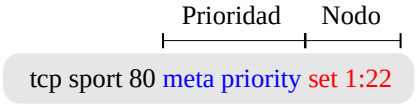
Parámetro	Descripción
parent [mayor:menor]	Nodo padre en la jerarquía
classid [mayor:menor]	Identificador único del nodo en la jerarquía
prio [prioridad]	Mapa de prioridades para el proceso *round robin* de reparto entre las clases hijas
rate [caudal]	Caudal garantizado para esta clase (y sus hijos)
ceil [caudal]	Caudal máximo alcanzable de envío para esta clase (y sus hijos)
burst [bytes]	Tamaño en bytes de la ráfaga que puede superar el caudal máximo
cburst [bytes]	Tamaño en bytes de la ráfaga que se puede emitir instantáneamente, es decir directo a la interfaz a máxima velocidad
quantum	Cantidad de bytes usados en el reparto *round robin* entre clases hijas de misma prioridad

Tabla 4. Parámetros admitidos en las clases de la disciplina HTB.

2.4. Clasificación de paquetes

La clasificación consiste en aplicar reglas a los paquetes para establecer un destino en el árbol jerárquico de disciplinas de la interfaz. Tradicionalmente estas reglas es establecían utilizando la herramienta *tc filter*. Esta herramienta trabaja con los paquetes a bajo nivel y es muy eficiente, en cambio, tiene poca documentación y tiene una sintaxis complicada. Por este motivo, en este laboratorio se propone realizar toda la clasificación con NFT, tal y como viene trabajando a lo largo del curso.

NFT soporta la clasificación de paquetes en la estructura de clases y colas creadas con la herramienta *tc*. La forma de hacerlo es usar la acción no final `meta priority set <hex-val>` que permite marcar un paquete para ser enviado a la clase correspondiente. La siguiente regla NFT clasifica los paquetes de originados por un servidor Web HTTP en la clase 1:22.



Las sentencias NFT de clasificación pueden añadir a cualquier cadena, de hecho, al ser no finales se pueden establecer antes de una acción final `accept`. El siguiente ejemplo muestra un firewall restrictivo (política a drop), en el cual se aceptan y se clasifican los paquetes DNS salientes en la clase 1:10.

```
table inet filter {
    chain output {
         type filter hook output priority filter; policy drop;
         udp dport 53 ip daddr { 8.8.8.8, 1.1.1.1 } meta priority set 1:10 accept;
         udp sport 53 ip saddr { 8.8.8.8, 1.1.1.1.} accept;
    }
}
```

Clasificar los paquetes junto con reglas de filtrado aumenta la complejidad, dificultando la depuración y mantenimiento de las reglas en caso de firewalls complejos. Una opción alternativa que aparece

repetidamente en la red es crear una cadena separada conde únicamente se realiza la clasificación de paquetes como se muestra en el siguiente ejemplo.

```
chain forward {
   type filter hook forward priority filter; policy accept;
   meta priority none ip saddr 192.168.0.0/16 meta priority set 1:10
   meta priority none ip daddr 10.0.0.0/8 meta priority set 1:20

   # Resto del tráfico no clasificado
   meta priority none meta priority set 1:2
   }
```

Esta segunda forma de clasificar los paquetes consiste en que todas las reglas comienzan comprobando que no han sido ya clasificados (condición `meta priority none`). Así, cada regla de la cadena clasifica un paquete no previamente clasificado y que cumpla el resto de condiciones, en cuando una regla lo clasifique las demás reglas ya no operan. Además, se añade una última regla con una única condición, que el paquete no ha sido clasificado, entonces se envía a la clase por defecto (1:2). Este procedimiento tiene dos deficiencias, la primera que es si hay una previa clasificación en otra cadena, esta ya no opera, y en segundo lugar, puede aparece una gran secuencia de reglas que se ejecuta en su totalidad (aunque NFT internamente las puede optimizar).

Finalmente, se propone un mecanismo que clasifique paquetes aunque ya estuvieran marcados, y que pare el procesamiento tras marcar un paquete. Esto último se puede conseguir mediante la acción final *accept* o *return* tal y como se muestra a continuación.

```
chain forward {
   type filter hook forward priority filter; policy accept;
   ip saddr 192.168.0.0/16 meta priority set 1:10 return;
   ip daddr 10.0.0.0/8 meta priority set 1:20 return;

   # Resto del tráfico no clasificado
   meta priority set 1:2 return;
   }
```

Hay una restricción importante para que la clasificación opere correctamente, en la clasificación sólo es posible enviar los paquetes a las clases hijas de disciplinas *classful*, aquellas que no disponen de clases hijas no deben ser nunca el destino de un paquete, ni tampoco, la raíz de una clase *classful*.

2.5. Estadísticas

Para depurar la política de control de tráfico configurada, la herramienta *tc* incluye un modo de operación con el que presenta estadísticas recopiladas por cada disciplina. Las estadísticas recopiladas representan dos tipos de valores: acumulados e instantáneos. Todas las disciplinas recopilan al menos el número de bytes y de paquetes a los que se les ha aplicado dicha disciplina. La recopilación de estadísticas no se produce sólo en las disciplinas, las clases en muchas ocasiones también recopilan información, siendo ésta muy útil para optimizar la configuración o detectar posibles errores.

El siguiente ejemplo muestra las estadísticas de las disciplinas existentes en una configuración de ejemplo

tras la ejecución del comando *tc*:

```
# tc -s -d qdisc show dev eth1

qdisc htb 1: root refcnt 2 r2q 10 default 99 direct_packets_stat 0 ver 3.17
    Sent 4539558 bytes 13203 pkt (dropped 0, overlimits 7115 requeues 0)
    backlog 0b 0p requeues 0
qdisc sfq 299: parent 1:99 limit 127p quantum 1514b flows 127/1024 divisor 1024 perturb 5sec
    Sent 707131 bytes 8123 pkt (dropped 0, overlimits 0 requeues 0)
    backlog 0b 0p requeues 0
qdisc pfifo 222: parent 1:22 limit 100p
    Sent 3603227 bytes 3555 pkt (dropped 0, overlimits 0 requeues 0)
    backlog 0b 0p requeues 0
qdisc sfq 223: parent 1:23 limit 127p quantum 1514b flows 127/1024 divisor 1024 perturb 10sec
    Sent 72894 bytes 193 pkt (dropped 0, overlimits 0 requeues 0)
    backlog 0b 0p requeues 0
```

Los valores estadísticos son fácilmente interpretables: bytes enviados, paquetes enviados y paquetes descartados. El valor *overlimits* indica la cantidad de veces que se podía haber enviado un paquete pero la disciplina rechazó el envío por sobrepasar sus límites establecidos, de hecho, sólo la disciplina HTB establece límites de caudal, por eso, en las estadísticas mostradas sólo pueden existir *overlimits* para HTB.

El siguiente comando *tc* muestra como salida las estadísticas recopiladas en las clases:

```
# tc -s -d class show dev eth1

class htb 1:2 root rate 600000bit ceil 600000bit burst 3000b/8 mpu 0b overhead 0b cburst
1599b/8 mpu 0b overhead 0b level 7
    Sent 4941114 bytes 14340 pkt (dropped 0, overlimits 0 requeues 0)
    rate 48040bit 20pps backlog 0b 0p requeues 0
    lended: 1990 borrowed: 0 giants: 0
    tokens: 599562 ctokens: 307890

class htb 1:99 parent 1:2 leaf 299: prio 1 quantum 5000 rate 80000bit ceil 600000bit burst
1600b/8 mpu 0b overhead 0b cburst 1599b/8 mpu 0b overhead 0b level 0
    Sent 769677 bytes 9020 pkt (dropped 0, overlimits 0 requeues 0)
    rate 7768bit 14pps backlog 0b 0p requeues 0
    lended: 8055 borrowed: 965 giants: 0
    tokens: 2279562 ctokens: 307890
class htb 1:21 parent 1:2 prio 0 quantum 1000 rate 25000bit ceil 100000bit burst 1600b/8 mpu
0b overhead 0b cburst 1600b/8 mpu 0b overhead 0b level 0
    Sent 95380 bytes 1023 pkt (dropped 0, overlimits 0 requeues 0)
    rate 192bit 0pps backlog 0b 0p requeues 0
    lended: 894 borrowed: 129 giants: 0
    tokens: 7400000 ctokens: 1850000

class htb 1:22 parent 1:2 leaf 222: prio 1 quantum 5000 rate 400000bit ceil 600000bit burst
1600b/8 mpu 0b overhead 0b cburst 1599b/8 mpu 0b overhead 0b level 0
    Sent 3987104 bytes 3949 pkt (dropped 0, overlimits 0 requeues 0)
    rate 38256bit 6pps backlog 0b 0p requeues 0
    lended: 3081 borrowed: 868 giants: 0
    tokens: 477500 ctokens: 318328
```

Los valores estadísticos mostrados en las clases HTB del ejemplo incluyen el valor instantáneo *rate* (caudal) expresado en *bits/seg* y *bytes/seg*. Son interesantes los valores *lended* y *borrowed* que representan los bytes prestados a las clases hijas y los bytes pedidos a la clase padre respectivamente.

2.6. Consideraciones adicionales

Adicionalmente para afinar en el cálculo de la latencia de los paquetes se debe profundizar en el funcionamiento interno del núcleo de Linux.

Según la documentación existente sobre la implementación de la capa de red en el núcleo de Linux, cuando un proceso del sistema envía un paquete, el paquete se pone en la disciplina de la interfaz para su envío.

Pero el controlador del medio físico incluye una cola adicional llamada *ring_buffer*, que se debe considerar. El núcleo se comunica únicamente con la disciplina raíz de la interfaz y solicita paquetes siempre que la cola en anillo del driver no esté llena. Sólo se obtienen paquetes de la disciplina raíz si hay hueco en el anillo de envío y el anillo no está parado, este comportamiento se ilustra en la figura 4.

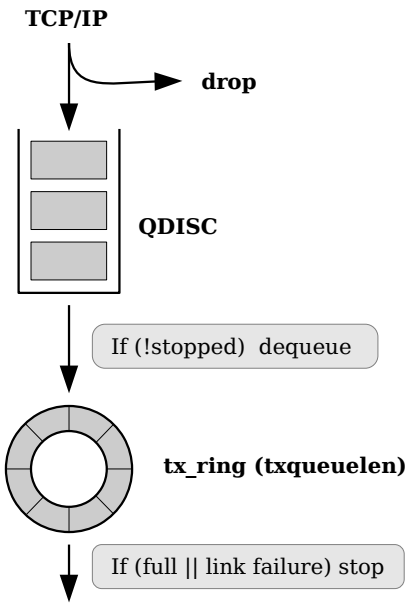


Figura 4. Colas del núcleo de Linux.

El tamaño del anillo de envío es independiente de la cola existente en la disciplina raíz y se puede considerar como una cola adicional inevitable. Este tamaño de anillo puede ser alterado siempre que el driver lo soporte y se puede consultar con el comando *ip link* (parámetro *txqueuelen*). Para cambiarlo sólo es posible mediante el comando *ip link* con la siguiente sintaxis: `ip link set eth0 txqueuelen 500`.

Otra consideración importante es el tamaño MTU cuando se configuran los parámetros de las disciplinas. Es importante ser consciente que muchos parámetros no deben establecerse por debajo de 1500 que es la MTU predeterminada. Por ejemplo, una disciplina TBF o HTB con una cubeta de tamaño inferior a este valor puede hacer que no se envíe ningún paquete, y si se tienen colas en bytes menores de este valor de MTU también puede ocurrir que no se encole ningún paquete.

Por último, considerando el tamaño de MTU es posible configurar una disciplina TBF o HTB para que se comporte como una cubeta con pérdidas (*Leaky bucket*), basta con que el tamaño de la cubeta sea la MTU para que no puedan ocurrir ráfagas.

3. Laboratorio guiado

Las pruebas a realizar en el laboratorio consistirán en realizar el conformado de tráfico para la red virtual desplegada a lo largo de los laboratorios de la asignatura. Se realizará todo el conformado en la máquina que hace de puerta de enlace (GW) simulando que se dispone de un ancho de banda limitado. Todo el control del tráfico se realizará sobre el tráfico saliente, ya que como es habitual, no se puede controlar el tráfico más allá de la frontera de red.

Para limitar este ancho de banda saliente se utilizarán las disciplinas descritas anteriormente, y a lo largo del bloque de laboratorio guiado se realizarán dos ejemplos, el primero consiste en probar diferentes disciplinas y observar los efectos sobre flujos en diferentes condiciones. La segunda parte guiada usará una solución de carácter general basada principalmente en la disciplina HTB donde se comprobará la flexibilidad en la

configuración de este tipo de disciplina.

Antes de comenzar se debe preparar el entorno de trabajo con algunos ficheros para poder realizar transferencias de datos por diversos protocolos. Para realizar todas las pruebas se requiere una máquina externa y es recomendable que esté en la misma red, no se recomienda realizar las pruebas desde el equipo local, ya que está conectado mediante una VPN desde una conexión externa con caudal inestable. Así, para este laboratorio se usará en todo momento la máquina virtual compartida.

3.1. Soluciones con múltiples disciplinas

Esta primera parte del laboratorio consiste en probar diferentes disciplinas siguiendo los ejemplos de las sucesivas tareas. Debe guardar todos los comandos *tc* utilizados en los ficheros de texto indicados para su posterior evaluación, recuerde que debe respetar la ubicación y los nombres de los ficheros.

Tarea 1.- Inicie una sesión de escritorio remoto (no SSH) en la máquina compartida para realizar las pruebas de transferencia desde ahí y no cierre la sesión. Simultáneamente en su máquina GW ejecute el comando indicado, como administrador, para mostrar estadísticas sobre la disciplina establecida para la interfaz de forma predeterminada (debe sustituir eth0 por el nombre de la interfaz externa, si es que difiere).

```
tc -s -d qdisc show dev eth0
```

T1.1.- Cree un directorio nuevo para alojar todos los ficheros de este laboratorio: `mkdir /root/qos`. Igual que en los laboratorios anteriores, debe usarse este directorio y mantener el nombre indicado en cada tarea para una correcta evaluación.

T1.2.- Para facilitar la obtención de estadísticas cree un fichero /root/qos/estadisticas.sh y añada el permiso de ejecución al mismo. El fichero mostrará de forma continuada las estadísticas en un terminal, siendo el contenido el siguiente:

```
#!/bin/bash
watch -n0 '
    tc -s -d qdisc show dev eth0
    echo "-- Clases -----------------------------------"
    tc -s -d class show dev eth0'
```

Código 1. Script para mostrar estadísticas de forma continuada.

T1.3.- Ejecute este script (/root/qos/estadisticas.sh) en un terminal SSH de GW y deje este terminal visible durante todas las pruebas realizadas posteriormente a lo largo de las tareas.

T1.4.- Para poder realizar pruebas de transmisión debe ubicar un fichero de al menos 20MBytes en tres de las ubicaciones configuradas en la sesión anterior de laboratorio: servidor Web/FTP interno, servidor Samba interno, y cuenta de usuario *tai* en GW, esta última es para realizar transferencias por SSH. Puede crear el fichero con el comando *truncate*, el siguiente ejemplo crea un fichero de 20MiB: `truncate -s 20M fichero-grande.dat`.

T1.5.- Ahora hay comprobar que los ficheros creados están ubicados correctamente y son accesibles desde la máquina compartida por diferentes protocolos: HTTP, FTP, y SSH. Esta prueba y todas las demás transferencias se realizan desde la máquina compartida. En primer lugar, para probar HTTP se usará el comando wget de la forma indicada en el código 2 y además, tras la ejecución, debería estar el *fichero-grande.dat* de 20Mbytes en el directorio actual (`ls -lh`), observe en la salida el caudal medio de la transferencia:

```
wget http://192.168.20.X/fichero-grande.dat

--2020-12-11 19:18:19--  http://192.168.20.X/fichero-grande.dat
Conectando con 192.168.20.X:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 1048576 (1,0M)
Grabando a: “fichero-grande.dat”

fichero-grande.dat     100%[=========================>]   1,00M  --.-KB/s    en 0,002s

2020-12-11 19:18:19 (112 MB/s) - “fichero-grande.dat” guardado [1048576/1048576]
```

Código 2. Comando para realizar transferencias HTTP.

T1.6.- Para probar el servicio FTP de su máquina VM1 se puede utilizar también *wget* escribiendo un URL correspondiente al protocolo FTP. Cuando *wget* usa el protocolo FTP, intenta conectar en modo pasivo al servidor, por ello, debe tener el modo pasivo correctamente configurado del laboratorio anterior. Ejecute la siguiente prueba desde la máquina compartida asegurándose que se ha transferido el fichero:

```
wget ftp://web:tai@192.168.20.X/html/fichero-grande.dat

--2020-12-11 19:48:29--  ftp://tai:*password*@192.168.20.X/fichero-grande.dat
           => “fichero-grande.dat.2”
Conectando con 192.168.20.X:21... conectado.
Identificándose como tai ... ¡Dentro!
==> SYST ... hecho.   ==> PWD ... hecho.
==> TYPE I ... hecho.  ==> no se necesita CWD.
==> SIZE fichero-grande.dat ... 1048576
==> PASV ... hecho.   ==> RETR fichero-grande.dat ... hecho.
Longitud: 1048576 (1,0M) (probablemente)

fichero-grande.dat. 100%[====================>]   1,00M  --.-KB/s    en 0,001s

2020-12-11 19:48:29 (112 MB/s) - “fichero-grande.dat” guardado [1048576]
```

Código 3. Comando para realizar transferencias FTP.

T1.7.- El mismo modo se debe preparar y probar el servicio Samba. Se usará la compartición publica creada en el laboratorio anterior ya que no requiere usuario y contraseña para descargar ficheros. Así, debe crear el archivo fichero-grande.dat de 20MiB en el directorio /srv/public de VM2 y probar la transferencia desde la máquina compartida con el siguiente comando:

```
smbget -n  smb://192.168.20.X/public/fichero-grande.dat -O

Using workgroup WORKGROUP, user tai
[fichero-grande.dat] 20,00MB of 20,00MB (100%) at 8,67MB/s
Downloaded 20,00MB in 2 seconds
```

Código 4. Comando para realizar transferencias Samba.

T1.8.- Por último, para probar una transferencia SSH puede usar el comando *sftp* o *scp*, ambos funcionan a través de un túnel SSH. El siguiente ejemplo se transfiere de nuevo *fichero-grande.dat* por SSH, pero debe crear este fichero en la cuenta *tai* de la máquina GW, ya el servicio SSH no está redirigido a ninguna máquina interna:

```
sftp tai@192.168.20.X:fichero-grande.dat

tai@192.168.20.X's password:
Connected to 192.168.20.X.
Fetching /home/tai/fichero-grande.dat to fichero-grande.dat
/home/tai/fichero-grande.dat                  100% 1024KB  90.3MB/s   00:00
```

Código 5. Comando para realizar transferencias SSH-SFTP.

Tras probar las transferencias y verificar que el fichero se descarga correctamente (usando los 4 protocolos), se procederá a probar diferentes disciplinas. En todas las tareas posteriores se volverán a realizar las transferencias y medir el caudal usando los tres comandos anteriores.

Tarea 2.- Con la primera disciplina se pretende limitar el caudal de salida de la interfaz exterior del entorno virtual a 500Kbit/s mediante una cubeta con fichas. Ésta disciplina correspondiente a TBF en Linux.

T2.1.- Cree un nuevo fichero `/root/qos/qos-t2.sh` con permiso de ejecución para ir añadiendo los comandos. Con el primer comando se establecerá una disciplina TBF como disciplina raíz de la interfaz, añada en el fichero los comandos indicados:

```
#!/bin/bash
set -x
tc qdisc del dev eth0 root
tc qdisc add dev eth0 root handle 1:0 tbf rate 300Kbit latency 50ms burst 1540
```

Código 6. Ejemplo de disciplina TBF.

T2.2.- Ejecute el script `/root/qos/qos-t2.sh` y observe los efectos en el terminal donde se muestran las estadísticas.

T2.3.- Desde la máquina compartida conecte al servidor Web y realice una transferencia mediante HTTP observando la tasa de transferencia, puede usar el comando `wget http://192.168.20.X/fichero-grande.dat`. ¿Corresponde con el caudal de 300Kbit/s establecidos?

T2.4.- Sin detener la transferencia HTTP realice otra transferencia simultánea mediante FTP. Puede

utilizar *filezilla* ya que muestra la tasa de transferencia o de nuevo *wget* siguiendo el ejemplo mostrado en el código 3 (pág. 18) ¿Se distribuye el caudal disponible equitativamente entre las dos descargas? ¿Y si añade una tercera transferencia mediante Samba?

Junto con el material de laboratorio se adjunta un script llamado *descargas.sh* pensado para realizar varias transferencias HTTP/FTP simultáneas. Para que opere correctamente debe usarlo desde el escritorio de una máquina Linux con los programas *pv*, *netcat*, *socat* y *xterm* instalados (puede instalarlos con APT). Se debe usar en las tareas posteriores y su uso consiste en ejecutarlo con dos argumentos según la siguiente sintaxis: `./descargas.sh N URL` donde N es el número de transferencias simultáneas y URL es la dirección completa del fichero. El URL puede ser tanto el ejemplo mostrado en el código 2 para HTTP o en el código 3 para FTP.

Tarea 3.- Usando el escritorio de la máquina compartida, descargue el script *descargas.sh* y añada el permiso de ejecución al mismo con el comando *chmod*. Ejecute el comando sin argumentos para que se muestre la ayuda de uso. Este comando y el resto debe realizarlo desde el escritorio remoto de la máquina compartida, no mediante SSH.

T3.1.- Desde el escritorio remoto de la máquina compartida se probarán tres transferencias simultáneas desde su GW. Ejecute el comando indicado utilizando su IP pública y se abrirán 3 ventanas donde debe observar la tasa de transferencia (considere que a veces las ventanas se superponen y tiene que moverlas):

```
./descargas.sh 3 http://192.168.20.X/fichero-grande.dat
```

T3.2.- Repita el comando anterior realizando simultáneamente 10 transferencias HTTP. Sin parar estas últimas, de manera simultánea intente acceder por FTP para descargar archivos, comprobará que las conexiones no funcionan correctamente.

T3.3.- De nuevo, sin detener ninguna transferencia intente acceder por SSH a su GW para obtener un terminal de texto de la máquina. ¿Responde correctamente el flujo interactivo SSH con las pulsaciones de teclas?

En la situación anterior ocurren varios efectos, el principal son las fluctuaciones en la tasa de cada transferencia. Se debe principalmente al descarte de multitud de paquetes de la cola de salida, de hecho, observe en las estadísticas los paquetes contados como dropped. El resultado observable habitualmente es que una de las transferencias se apodera de prácticamente todo el caudal disponible dejando detenidas las demás.

Tarea 4.- Se realizará otra mejora creando un árbol de disciplinas jerárquico. Se utilizará la disciplina PRIO con tres bandas de prioridad con el objetivo de dar prioridad al servidor WEB.

T4.1.- Copie el *script* anterior qos-t2.sh con el nombre qos-t4.sh para añadir una nueva línea con el comando:

```
tc qdisc add dev eth0 parent 1:0 handle 10: prio
```

T4.2.- Ejecute el nuevo script (qos-t4.sh) y compruebe, usando el terminal con las estadísticas, si se han añadido correctamente las disciplinas. Deben existir 3 bandas creadas automáticamente, asegúrese que la configuración mostrada corresponde a la figura 5.

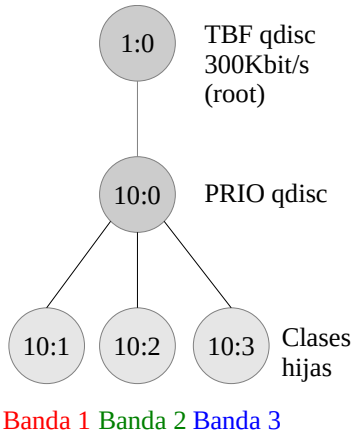


Figura 5. Disciplina PRIO.

T4.3.- Para clasificar los paquetes se utilizará *nftables*, pero no se utilizará un fichero separado, se usará una sección en el mismo fichero *bash-script* que admita la sintaxis NFT. Use el siguiente código para reemplazar completamente el contenido del fichero qos-t4.sh:

```
#!/bin/bash
set -x
tc qdisc del dev eth0 root
tc qdisc add dev eth0 root handle 1:0 tbf rate 300Kbit latency 50ms burst 1540
tc qdisc add dev eth0 parent 1:0 handle 10: prio

nft -f - << FIN

table ip qos
delete table qos

table ip qos {
    chain postrouting {
        type filter hook postrouting priority 0; policy accept;
        tcp sport 80 meta priority set 10:1 counter return
        meta priority set 10:3 counter return
    }
}

FIN
```

T4.4.- Fíjese que en las reglas NFT anteriores se han clasificado los paquetes HTTP en la banda 1 el resto del tráfico en la banda 3. Realice una única transferencia HTTP y observe (si es posible) en el terminal SSH con estadísticas cómo los paquetes HTTP se están clasificando en la clase 10:1. Debe observar los contadores de la tres bandas. ¿Por qué el terminal SSH con las estadísticas que queda congelado?

T4.5.- Pare la transferencias HTTP y espere hasta recuperar la conexión en el terminal SSH de GW. ¿En qué banda se clasifican los paquetes SSH?

En el ejemplo mostrado la cola prioritaria anula completamente el resto de flujos menos prioritarios, la planificación es estricta, mientras una cola prioritaria tenga paquetes encolados, sólo se extraen paquetes de esta cola. Otro aspecto importante que se debe tratar es el modo en que se aplican las reglas. En el esquema

de Netfilter el gancho *postrouting* es el último aplicarse una vez superado tanto *output* o *forward*. En este lugar se puede clasificar cualquier paquete saliente, provenga tanto de GW como de la red interna. Considere que en el ejemplo propuesto se ha usado la acción *return*, la cual, abandona el procesado de la cadena en ese punto. Se usa por claridad, pero también serviría *accept* que aceptaría el paquete tampoco seguiría procesando la cadena.

Tarea 5.- Ahora se propone crear reglas para dar prioridad a los flujos SSH, ya que suelen ser interactivos y habitualmente no consumen gran ancho de banda. Copie el fichero qos-t4.sh como qos-t5.sh y realice dos cambios indicados a continuación:

T5.1.- Cambie la regla NFT para que los flujos HTTP se clasifiquen en la clase 10:2. Añada una nueva regla para que los flujos SSH se clasifiquen en la clase 10:1. Finalmente, el resto del tráfico debe clasificarse en la clase 10:3.

T5.2.- Ejecute el nuevo script qos-t5.sh y realice 10 transferencias simultáneas HTTP. Al mismo tiempo compruebe que sigue respondiendo el servidor SSH de manera interactiva en un terminal. Revise el terminal de estadísticas para comprobar que la clasificación es correcta. ¿Afecta el uso de SSH a las transferencias HTTP?

T5.3.- Sin detener las 10 descargas proceda a transferir por SSH el fichero de gran tamaño mediante el comando *scp* o *sftp* siguiendo el ejemplo del código 5 (pág. 19) ¿Deja el terminal SSH de responder interactivamente? ¿Siguen operando correctamente las transferencias HTTP simultáneas?

T5.4.- Sin detener ninguna transferencia, intente bajo estas circunstancias usar una de las máquina internas, por ejemplo VM3, para conectarse a Internet. Desde VM3 actualice el listado de los paquetes con `apt update`, o navegue con Firefox, seguramente observará que toda la red interna ha perdido la conexión con la red exterior, el enlace de salida está saturado por la transferencia SSH-SFTP y las HTTP.

Tras analizar las pruebas realizadas en las últimas tareas han surgido los siguientes problemas:

- Cuando varios flujos compiten en una cola siempre alguno de ellos es dominante sobre los demás, esto se ha visto al realizar multitud de transferencias simultáneas HTTP, todas suelen detenerse menos una.
- Los flujos clasificados en la cola de mayor prioridad anulan el resto de colas, es decir, acaparan todo el caudal disponible. Esta situación empeora en un ataque DoS hacia el servicio prioritario: se anularían todos los flujos, incluso queda fuera de servicio la red interna al no poder ni resolverse las solicitudes DNS.

Para atajar en la medida de lo posible esta situación se ampliará el árbol de disciplinas en las siguientes tareas.

Tarea 6.- En primer lugar se intentarán equilibrar los caudales de las transferencias simultáneas. Se añadirá una política SFQ en el árbol para tratar los flujos de la clase 10:3 tal y como se muestra en la figura 6. Esta política es similar a WFQ explicada en teoría pero aplicada a flujos detectados dinámicamente.

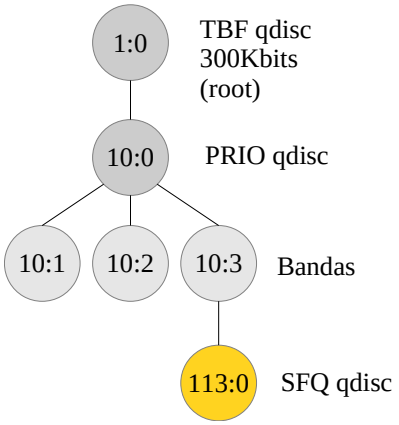


Figura 6. Disciplina SFQ.

T6.1.- De nuevo copie el fichero qos-t5.sh como qos-t6.sh y añada a este último la nueva disciplina.

```
tc qdisc add dev eth0 parent 10:3 handle 113: sfq
```

T6.2.- Pruebe desde la máquina compartida 3 transferencias FTP simultáneas. Observe si todos los flujos adquieren ancho de banda de una manera más equitativa. ¿Por qué se han usado transferencias FTP en vez de HTTP?

T6.3.- Observe en la máquina GW la ventana de estadísticas, deberán ir apareciendo clases hijas dinámicamente en el nodo 10:3 por cada uno de los flujos detectados. Cuando paren las transferencias estas clases desaparecerán.

Tarea 7.- Vuelva a copiar el script de la tarea anterior como qos-t7.sh, y establezca las reglas NFT adecuadamente con el objetivo de conseguir la clasificación indicada en la figura 7, clasificando el resto del tráfico en la clase 10:2.

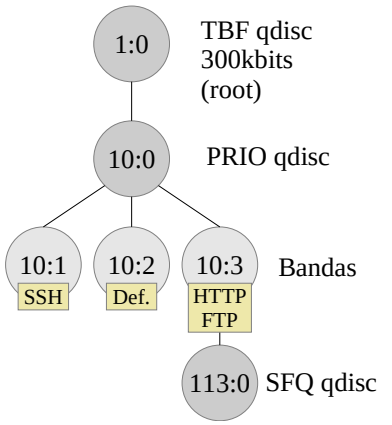


Figura 7. Ubicación de los flujos.

T7.1.- Ejecute una única transferencia FTP mediante `./descargas.sh 1 ftp://web:tai@192.168.20.21/html/fichero-grande.dat` y revise que los paquetes pasan por la clase 10:3. Si no funciona, considere que para FTP necesitará varias reglas NFT.

T7.2.- Ejecute 2 transferencias simultáneas HTTP, y una vez iniciadas, comience otras 2 transferencias FTP simultáneas y revise que se crean dinámicamente las clases SFQ equilibrando las descargas. Compruebe si el servicio SSH sigue operando con prioridad en estas circunstancias.

T7.3.- Detenga la transferencias anteriores e inicie 10 transferencias HTTP verificando que las máquinas internas no han perdido la conexión a Internet. ¿Por qué ahora la sobrecarga HTTP no afecta a la red interna? ¿En qué banda se clasifican las transferencias Samba?

La solución anterior todavía presenta algunos problemas ya que pueden ocurrir situaciones donde el servidor HTTP se quede sin servicio, principalmente si la clase 10:2 se satura debido al tráfico de la red interna. En la configuración de colas prioritarias anterior, siempre existe la posibilidad de que la cola más prioritaria anule totalmente las transferencias de las menos prioritarias.

Tarea 8.- En primer lugar se pretende evitar que la cola 10:2, anule el tráfico HTTP/FTP. Así, se le añadirá una disciplina TBF con un caudal máximo inferior al disponible para que siempre quede caudal para la clase

menos prioritaria.

T8.1.- De nuevo copie el fichero de la tarea anterior como qos-t8.sh para trabajar en él. Añada una disciplina TBF en el árbol hijo de 10:2 de la siguiente forma:

```
tc qdisc add dev eth0 parent 10:2 handle 112: \
   tbf rate 150kbit limit 10k burst 3000
```

T8.2.- Ejecute el script con la nueva disciplina y, en este caso, realice una descarga por el protocolo SMB (vea código 4 pág. 19) revisando que se clasifica en 10:2 y que opera con menos de los 300Kbits disponibles.

T8.3.- Sin parar la transferencia anterior, inicie una transferencias HTTP o FTP simultáneamente y compruebe que opera con el caudal sobrante que no es consumido por la banda 10:2. ¿Qué ocurre si en esta situación se transfiere por SSH/SFTP un fichero de gran tamaño?

Tarea 9.- Termine la solución en un fichero llamado qos-t9.sh siguiendo el esquema de figura 8. En este caso se han aumentado los caudales, se clasifican los paquetes DNS (protocolo UDP) y se añade otra disciplina SFQ a la cola prioritaria.

T9.1.- Una vez implementada realice 20 ó 30 transferencias HTTP simultáneas ¿se equilibra el caudal de las descargas?.

T9.2.- Pare las descargas anteriores e inicie una descarga SMB. Inicie y pare descargas HTTP comprobando que operan correctamente y que el caudal SMB es fijo, no se ve alterado nunca con las descargas HTTP.

En el ejemplo anterior se ha podido comprobar lo estricto que llegan a ser las políticas QoS de colas prioritarias. La banda 10:2 dispone de 800Kbit/s y aunque no existan flujos adicionales, los 200Kbit/s restantes nunca de aprovechan.

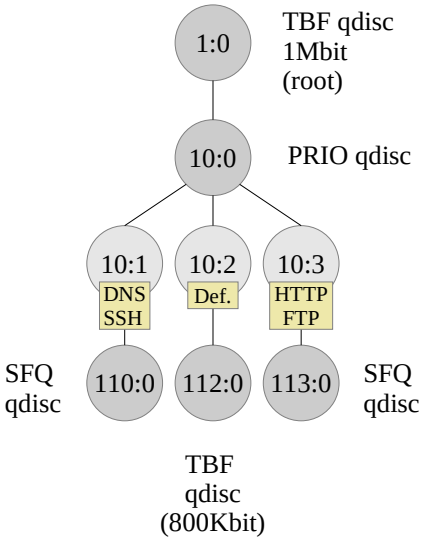


Figura 8. Ejemplo con múltiples disciplinas.

Tarea 10.- En este último ejemplo se realizará una simulación de un enlace saturado, para ello, copie el fichero anterior como qos-t9.sh como qos-t10.sh y en este nuevo fichero reduzca el caudal raíz a 300Kbit y el de la banda 10:2 a 150Kbits.

T10.1.- Realice 30 transferencias HTTP verá que todas las transferencias operan de manera irregular, algunas se paran y el caudal no se equilibra.

T10.2.- Pare las descargas y en la máquina que ejecuta el servidor HTTP (VM1) cambie la MTU a 100 mediante `ip link set dev eth0 mtu 100`. Repita las 30 descargas y observe el resultado ¿Por qué ahora si se equilibran las descargas?.

En el último caso mostrado se disminuye el tamaño MTU y el efecto inmediato es una disminución considerable de la eficiencia de los protocolos al aumentar la relación entre los tamaños de las cabeceras y

los datos enviados de cada paquete. En cambio, aumenta la interactividad y la fluidez al enviarse más paquetes por segundo. Para comprender lo ocurrido considere que en el ejemplo se dispone de 300Kbit/s y se deben repartir entre 30 flujos HTTP siendo la MTU de 1500bytes, por tanto, los paquetes son de este tamaño. Haciendo el siguiente cálculo:

$$300\,Kbits/seg = 37500\,Bytes/seg$$

$$\frac{37500\,Bytes/seg}{1500\,Bytes} = 25\,paquetes/seg \quad \Rightarrow \quad \frac{25\,paquetes/seg}{30\,flujos} \simeq 0.8\,paquetes/seg$$

El resultado muestra que cada flujo apenas recibe más de 1 paquete por segundo, llevando esta situación al extremo se podrían agotar los tiempos de espera de los paquetes TCP. Disminuyendo la MTU a 100Bytes tal y como se hizo en T10.2.- se aumenta el número de paquetes por segundo que recibe cada flujo, llegando a alcanzar 12 paquetes por segundo y aumentando la fluidez. Este procedimiento aumenta la interactividad de un flujo aunque se pierda eficiencia, pero hay que ser cuidadoso al aplicarlo ya que genera otro tipos de problemas de red, ya que pueden descartarse paquetes con MTU superior.

El caso mostrado como ejemplo llega a complicarse si se aumenta el número de servicios y el número de bandas de prioridad. Llega a ser complicado el cálculo de los tamaños de cola, latencias y caudales para cada uno de los nodos. Como se indicó anteriormente, esta solución no aprovecha bien el caudal disponible, la banda 10:2 tiene un caudal estricto y no puede solucionarse con este tipo de disciplinas.

Por los motivos expuestos se propone el estudio de soluciones basadas en la disciplina HTB que mejora la eficiencia.

3.2. Implementación con HTB

HTB simplifica el procedimiento de clasificación y aprovecha en mayor medida el ancho de banda disponible. La configuración de este tipo de disciplina es sencilla, de hecho, la página de manual de la misma (`man tc-htb`) es muy reducida ya que, aunque tiene pocos parámetros, éstos permiten muchas posibilidades de configuración. En los ejemplos propuestos se verá como es posible implementar diferentes disciplinas usando únicamente HTB con diferentes configuraciones.

Tarea 11.- En primer lugar se establecerá como disciplina raíz de la interfaz una disciplina HTB. Cree un nuevo fichero `qos-t11.sh` para almacenar la configuración y añada el siguiente contenido:

```
#!/bin/bash
set -x
tc qdisc del dev eth0 root
tc qdisc add dev eth0 root handle 1:0 htb default 1
tc class add dev eth0 parent 1:0 classid 1:1 htb rate 1Mbit burst 1500
```

T11.1.- Realice una transferencia HTTP y pruebe si opera correctamente el límite establecido. Después, realice 10 transferencias simultáneas HTTP para comprobar si se equilibra la velocidad entre diferentes descargas o funcionan irregularmente como en casos anteriores.

T11.2.- Use la ventana de estadísticas para ver como HTB muestra los paquetes y los tokens (fichas)

además de otros datos.

En el ejemplo anterior se ha creado una disciplina raíz del tipo HTB en la interfaz, pero es obligatorio indicar cual es la clase hija donde se encolará de manera predeterminada el tráfico, para el ejemplo, corresponde a `default 1`, indicado en rojo. El número indicado en el parámetro *default* es el número menor de la clase hija destino, es decir `default 1` clasifica los paquetes a la clase hija `1:1`. La configuración HTB realizada en el ejemplo es equivalente a la realizada en la sección anterior mediante TBF para limitar todo el caudal de la interfaz.

Ahora se presenta la estructura jerárquica de HTB y el proceso *Round-Robin* seguido en el reparto del caudal. Se propone realizar una configuración usando únicamente en HTB que intente asemejarse a una disciplina PRIO (colas de prioridad). Observe el parecido entre la figura 5 (pág. 21) y lo mostrado en la figura 9. En esta última se establecen prioridades a las clases HTB consiguiendo el un efecto parecido a una disciplina PRIO, pero presenta algunas diferencias que se observarán a continuación en el ejemplo usado.

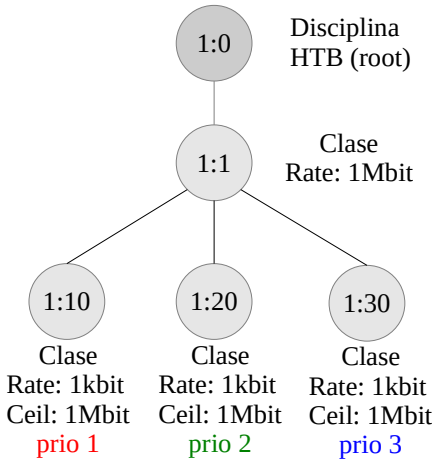


Figura 9. Jerarquía HTB equivalente a PRIO.

En primer lugar, se utilizará la característica de caudal mínimo/máximo de HTB y se observará como la clase padre distribuye ancho de banda entre los hijos. Para comprender el siguiente ejemplo, recuerde que el parámetro *rate* establece el caudal garantizado y el parámetro *ceil* es el máximo caudal alcanzable por la clase. En el proceso de reparto de caudal entre los hijos, la prioridad establece quien es el primer hijo que recibe el ancho de banda sobrante, y cuando no quede caudal sobrante, los menos prioritarios no reciben nada.

Tarea 12.- Para implementar la jerarquía mostrada en la figura 9 use un nuevo fichero llamado qos-t12.sh, y como antes, copie qos-t11.sh para usarlo como punto de partida.

T12.1.- En el nuevo fichero establezca para la disciplina HTB raíz como clase predeterminada la 1:30, es decir cambie `default 30` y añada tres clases hijas a 1:1 mediante el código indicado:

```
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 1kbit ceil 1Mbit prio 1
tc class add dev eth0 parent 1:1 classid 1:20 htb rate 1kbit ceil 1Mbit prio 2
tc class add dev eth0 parent 1:1 classid 1:30 htb rate 1kbit ceil 1Mbit prio 3
```

T12.2.- Ahora añada al fichero la clasificación NFT con una única regla, de forma que, se clasifique todo el tráfico HTTP hacia la clase 1:10. Con esta disciplina no necesita una regla final que clasifique por defecto el tráfico, ya lo hace HTB.

T12.3.- Desde la máquina compartida inicie una única transferencia HTTP y en la ventana de estadísticas verifique que la regla NFT anterior opera correctamente, si es así, verá en las estadísticas como se incrementan los bytes enviados de la clase htb 1:10, pero el efecto en el termina SSH es el mismo que con colas de prioridad, se congela el terminal.

T12.4.- Si compara el ejemplo anterior con el realizado anteriormente pero con las disciplinas PRIO (T4.5.-) el resultado es diferente. Con las colas de prioridad se anulaba SSH completamente, pero en este caso, no se anula del todo. Es debido a que las clases tienen un caudal mínimo de 1kbit (*rate 1kbit*). Para conseguir el mismo efecto que con PRIO habría que poner *rate* a cero ¿Qué ocurre si intenta poner *rate* a cero en el fichero qos-12.sh? ¿Cual es el mínimo posible?

T12.5.- Como última prueba, realice 20 o más transferencias HTTP para observar el desequilibrio en los caudales. Soluciónelo añadiendo una disciplina SFQ a la clase 1:10.

En el ejemplo mostrado se ha conseguido un efecto parecido, pero no igual, que el de las colas de prioridad. Se ha utilizado el mínimo *rate* permitido para las clases hijas, y se ha establecido adecuadamente el parámetro *prio* de cada una de las clases. El parámetro *prio* hace referencia a la prioridad en el reparto del caudal disponible desde la clase padre y el parámetro *ceil* indica el máximo caudal que una clase toma cuando la clase padre reparte caudal. Con todo esto, el funcionamiento se puede resumir como sigue:

La clase padre 1:1 dispone de un caudal de 1Mbit para repartir entre las clases hijas. Cuando las clases hijas solicitan caudal, pueden solicitar a la clase padre hasta 1Mbit (parámetro *ceil 1Mbit*). El algoritmo de reparto de la clase padre es el siguiente:

1. Se ordena por prioridad a todos los hijos que solicitan caudal.
2. Al más prioritario se le asigna todo el caudal solicitado sin sobrepasar el caudal disponible (1Mbit).
3. Si queda caudal disponible (no se ha sobrepasado 1Mbit), se vuelve aplicar el paso 1 con los hijos restantes y se hace el reparto de nuevo del paso 2 con el caudal disponible. Y así hasta agotar el caudal disponible.

Aplicando el algoritmo anterior al ejemplo de la figura 9, al solicitar el primer hijo caudal, puede quedarse con todo el caudal disponible del padre (1Mbit) ya que es el más prioritario (*prio 1*). Así, el resto de hijos sólo tienen 1kbit de ancho de banda.

La configuración mostrada no es equivalente a las colas de prioridad puesto que no es posible establecer el parámetro *rate* a cero en HTB. Además, cada clase HTB tiene parámetros que controlan las ráfagas (*burst* y *cburst*) que provocan otro efectos, en conexiones interactivas como SSH, donde un terminal SSH responde inicialmente a las pulsaciones hasta agotar los tokens/bytes de ráfaga y deja de responder totalmente tras varias pulsaciones de teclas continuadas.

HTB está pensado para garantizar caudal a todas las clases, ese es el principal motivo por el que no permite establecer *rate* a cero. En este sentido, el uso habitual de HTB consiste en dotar a cada una de las clases de un caudal garantizado (*rate*) suficiente para que el servicio, aunque lento, siga operando. Siguiendo este principio se propone usar una configuración razonable para mejorar considerablemente el funcionamiento del ejemplo mostrado anteriormente y, persiguiendo también el objetivo de aprovechar el ancho de banda completo disponible.

Tarea 13.- Siguiendo el esquema de la figura 10 y trabajando en un nuevo fichero de nombre qos-t13.sh cambie los parámetros *rate* de las tres clases 1:1X y establezca el parámetro *ceil* al valor indicado.

T13.1.- Clasifique todo el tráfico siguiendo la figura y considere que la clase predeterminada 1:20 se configura mediante HTB y no con reglas NFT.

T13.2.- Verifique que Samba se clasifica correctamente usando el comando *smbget* tal y como se usó en T1.7.-

T13.3.- Realice 20 transferencias simultáneas HTTP pruebe si la conexión SSH responde de manera interactiva.

T13.4.- Anule las 10 transferencias y realice una única transferencia HTTP para medir el caudal. Al mismo tiempo realice otra transferencia FTP. ¿Garantiza la clase 1:30 100kbits al tráfico HTTP? ¿Cuál es el caudal teórico que debe usar la conexión FTP? ¿Coincide el caudal teórico disponible para FTP en esta situación?

Comparando esta solución HTB con la solución multidisciplinar de la sección anterior se llega a la conclusión que HTB ofrece mayor facilidad y flexibilidad de configuración en su uso y configuración.

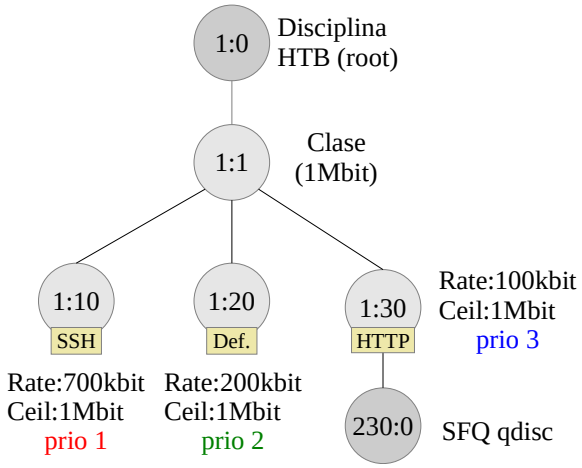


Figura 10. Jerarquía HTB con disciplinas adicionales.

4. Estudio no guiado

Para finalizar el laboratorio se propone realizar un ejercicio no guiado usando una jerarquía con la disciplina HTB. Concretamente, el ejemplo se muestra en la figura 11 y corresponde a una posible configuración para un caudal de salida de un servidor con diferentes servicios, donde sólo se controla la frontera de red, es decir, sólo se controla el caudal saliente.. Realice la implementación según las indicaciones siguientes.

Tarea 14.- Implemente la jerarquía HTB mostrada en la figura 11 en un nuevo fichero `/root/qos/qos-t14.sh` considerando lo siguiente:

T14.1.- La clase 1:1 e hijas debe incluirla, pero no se usarán en esta tarea, se usará posteriormente.

T14.2.- En la regla NFT para DNS considere que el servicio DNS se refiere a la peticiones DNS realizadas a servidores externos.

T14.3.- El la regla NFT para el protocolo SMTP se refiere a todos los paquetes salientes desde el puerto 25.

T14.4.- La clase 1:99 es la clase predeterminada para todo el tráfico restante.

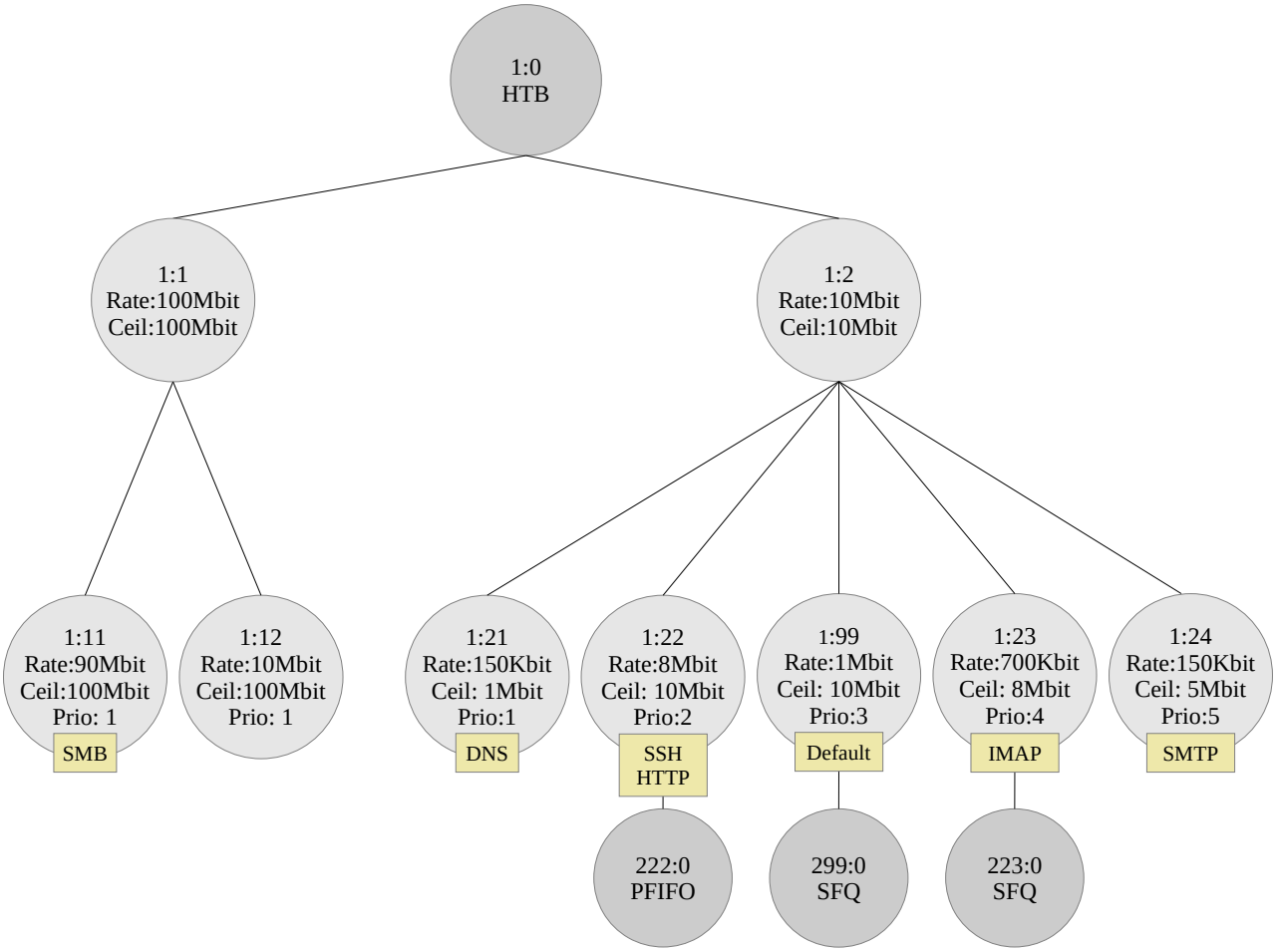


Figura 11. Ejemplo de jerarquía HTB con múltiples servicios.

Para poder testar las disciplinas desde la máquina compartida o máquina local, no es necesario instalar servicios adicionales a los usados en la asignatura. Para testar el ancho de banda de los puertos correspondientes a IMAP, SMTP, etc se ha preparado un *script* que hace uso de *netcat* en modo escucha, de forma que, cuando recibe una conexión entrante envía un flujo de datos continuo que no tiene fin. En la siguiente tarea se propone usar este mecanismo para testar si la tarea anterior se ha realizado de forma correcta.

Tarea 15.- Para realizar las pruebas se usarán los scripts *escucha-puerto.sh* y *descargas2.sh*. Disponibles en la página Web del material de laboratorio. El primero se usará en la máquina GW y el segundo en la máquina compartida o local.

T15.1.- Instale el paquete *socat* con *apt* en la máquina GW y pruebe el script *escucha-puerto.sh* en un terminal, de forma que se ponga a esperar conexiones en el puerto de correo electrónico (SMTP 25).

T15.2.- Desde la máquina compartida o su máquina local, conecte con usando el comando `nc 192.168.20.X 25` y vea lo que recibe por el terminal. Este flujo de datos es infinito así que debe usar CTRL+C para cortar la conexión.

T15.3.- También desde la máquina compartida o local ejecute el script *descargas2.sh* con varias

descargas simultáneas para testar el puerto de correo SMTP.

T15.4.- Use adecuadamente el script *escucha-puerto.sh* para testar también el puerto IMAP.

T15.5.- Con la configuración utilice *descargas.sh* y *descargas2.sh* masivamente hacia diferentes puertos y compruebe que opera correctamente su configuración.

T15.6.- En la situación previa de estrés asegúrese que la máquina VM1 y VM2 son capaces de resolver peticiones DNS desde la red interna sin ninguna demora, y que pueden también actualizar la lista de paquetes o hacer instalaciones de software con APT.

En la tarea anterior no se usó la clase 1:1 la cual tiene asignada un caudal de 100Mbi/ts. Ésta clase está pensada para dar servicio con mayor ancho de banda a toda la red virtual 192.168.20.0/24. Considere para esta situación que la máquina compartida está en la red 192.168.20.0/24, por lo que deberá funcionar a 100Mbit/s. En cambio el, equipo local (sea por VPN o el de laboratorio) deberá operar con la configuración de la tarea anterior de 10Mbit/s. Dada esta configuración, en esta la nueva tarea propuesta hay que hacer las pruebas desde la máquina compartida y también desde el equipo local y comparar resultados.

En esta última configuración se propone avanzar en NFT añadiendo una nueva cadena enlazada al mismo gancho (*postrouting*) pero con una prioridad diferente. La prioridad de las cadenas NFT no ha sido tratada todavía, pero es fácil de entender. Dadas varias cadenas de un mismo gancho (hook) se aplica de la siguiente forma:

- La prioridad es un número entero que puede ser negativo.
- Algunas valores de prioridad tienen asignado un nombre: *dnat=-100*, *filter=0*, *srcnat=100*, etc.
- Se ordenan las cadenas por prioridad creciente y se aplican las reglas.
- Recuerde que cualquier acción final *drop* descarta el paquete y se para todo el procesamiento del resto de cadena
- Recuerde que cualquier sentencia *accept* o *return* para el procesamiento en la cadena y pasa a la siguiente cadena (en orden de prioridad)

Tarea 16.- Cree un nuevo script vacío llamado /root/qos/qos-t16.sh de forma que incluya el script de la tarea anterior. El contenido podría ser de la siguiente forma:

```
#!/bin/bash

# Esta línea incluye otro fichero
source /root/qos/qos-t14.sh

# A partir de aqui se pueden añadir sentencias adicionales

nft -f - << FIN

# Se añadirá una cadena que se ejecute despues de la cadena "quos"
# estableciendo en el gancho la prioridad adecuada

table ip qos-prioritaria
delete table qos-prioritaria

table ip qos-prioritaria {
    chain postrouting {
        type filter hook postrouting priority 10; policy accept;
        # A partir de aquí se pueden volver a marcar paquetes ya marcados anteriomente
    }
}
FIN
```

T16.1.- Tal y como se muestra en el comentario del código anterior, se pueden añadir reglas NFT que sobrescriban la prioridad de los paquetes. Añada al final nuevas reglas que clasifiquen todo el tráfico hacia la red 192.168.20.0/24 de la siguiente forma:

- Todos los protocolos de Windows (Samba) en la clase 1:11
- El resto de paquetes con destino esta red en la clase 1:12

T16.2.- Ahora, para verificar su funcionamiento, compruebe el caudal de una transferencia HTTP desde la máquina compartida y las transferencias Samba verificando que ambas pasan por las clases correspondientes.

T16.3.- Revise que para el resto de redes se sigue limitando el caudal. Para ello, desde su equipo local realice una transferencia SMTP y conecte por SSH al mismo tiempo para ver en las estadísticas si la clasificación es correcta.

T16.4.- Considere que debe añadir un servicio VPN considerado como prioritario ¿A qué clase lo añadiría?, añada una regla NFT que lo clasifique según su decisión.

Tarea 17.- Opcional-5a: Puede realizar algunas pruebas opcionales no estudiadas en este laboratorio

T17.1.- Experimente con la disciplina CHOKE colapsando algún servicio. ¿Para qué sirve esta disciplina?

T17.2.- Compruebe que SFQ no es buena política para páginas WEB, si la página tiene mucho contenido (muchas imágenes) ocurren efectos extraños, se cargan todas la imágenes en paralelo y lentamente. Puede probar una simple PFIFO con un tamaño de un solo paquete.

T17.3.- Pruebe alterar el tamaño de cola de la interfaz para ver si afecta a los resultados, establezca valores grandes y pequeños con el comando *ip* y altere el tamaño de ráfaga en la clase 1:2.