DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorios de Tecnologías Avanzadas de la Información

Paulino Ruiz de Clavijo Vázquez <paulino@dte.us.es>

Rev. 2.9

Índice de contenido

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 1
Preparación del entorno de laboratorio

Enunciados de Prácticas de Laboratorio
Tecnologías Avanzadas de la Información

1. Introducción y objetivos

La duración estimada de esta sesión de laboratorio es de **2 horas**. El propósito general de esta sesión de laboratorio es la preparación de un entorno de trabajo basado en virtualización para ser utilizado a lo largo del curso. Los principales objetivos se resumen como sigue:

- Instalación y 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.

Para la realización de los laboratorios se facilitarán diversos ficheros, concretamente la tabla 1 resume el contenido y el objetivo .

Nombre del fichero	Descripción
linux-tai.ova	Ubuntu 12.04 Server preinstalado

Tabla 1. Ficheros necesarios durante la sesión de laboratorio.

2. VirtualBox

Virtualbox es una de las muchas soluciones de virtualización existentes y actualmente es propiedad de Oracle. Existe una versión con licencia GPL denominada Virtualbox OSE (*Open Source Edition*), que

se puede utilizar libremente. Durante este curso se utilizará este software para crear entornos virtuales de diversos equipos conectados a diferentes redes, y con ellos se realizarán todas las sesiones de laboratorio posteriores.

Para ahorrar el tiempo necesario para la instalación de un sistema operativo en cada una de las máquinas virtuales necesarias, se utilizarán las opciones de importación y exportación de Virtualbox, que facilitan la clonación de instalaciones completas de máquinas virtuales.

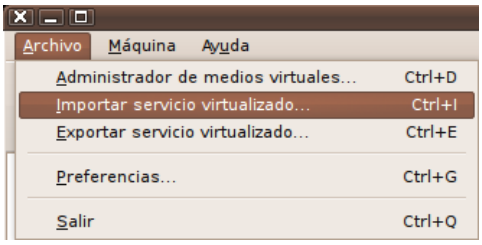

Figura 1. Importación de servicio virtualizado.

Se realizará la importación de una máquina linux (distribución Ubuntu) preinstalada mediante la opción de importación de servicio virtualizado.

Tarea 1.- Descarge el fichero *linux-tai.ova* en su disco local. Una vez descargado inicie la aplicación Virtualbox e importe la máquina virtual utilizando la opción de menú mostrada en la figura 1.

T1.1.- En la ventana de opciones de importación mostrada en la figura 2 es importante que renombre la máquina como *vbox-gateway-XYZ* donde XYZ son las iniciales del alumno y marque la casilla Reinicializar la dirección MAC de todas las tarjetas de red.

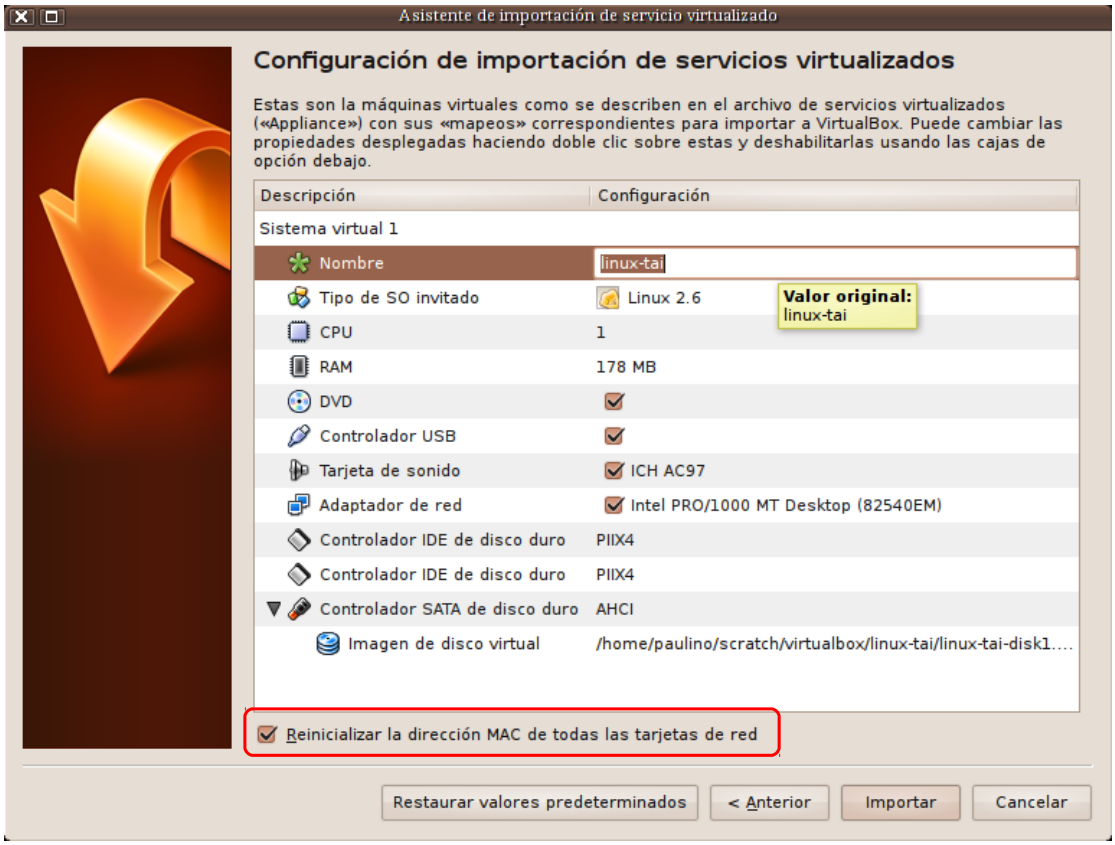

Figura 2. Asistente de importación de servicio virtualizado.

T1.2.- Inicie la nueva máquina para comprobar el correcto funcionamiento del sistema operativo. Use el usuario y la contraseña indicada.

T1.3.- Pruebe apagar la máquina desde del menú Máquina → Apagado ACPI.

Tarea 2.- Tras la importación de la primera máquina se realizarán clonaciones.

T2.1.- Seleccione la máquina importada anteriormente y usando el menú Máquina → Clonar, aparecerá un asistente donde debe indicar estas opciones: nombre de máquina a *vbox1*, casilla reiniciar direcciones MAC marcada y tipo de clonación a enlazada.

T2.2.- Repita los pasos anteriores y consiga una tercera máquina llamada *vbox2*.

T2.3.- Inicie las tres máquinas y compruebe si funcionan correctamente. Si se demora el arranque de alguna de ellas puede ser por la configuración de red, la cual, se realizará posteriormente.

2.1. Configuración de la red de máquinas virtuales

Tras disponer de tres máquinas virtuales diferentes, el objetivo es conseguir una configuración de red como la mostrada en la figura 3. Las tres máquinas estarán conectadas a una red virtual interna (192.168.0.0/24) y una de ellas hará de gateway, para ello, dispondrá de dos interfaces de red, una conectada a la red virtual interna y otra conectada a la red de laboratorio (192.168.20.0/24).

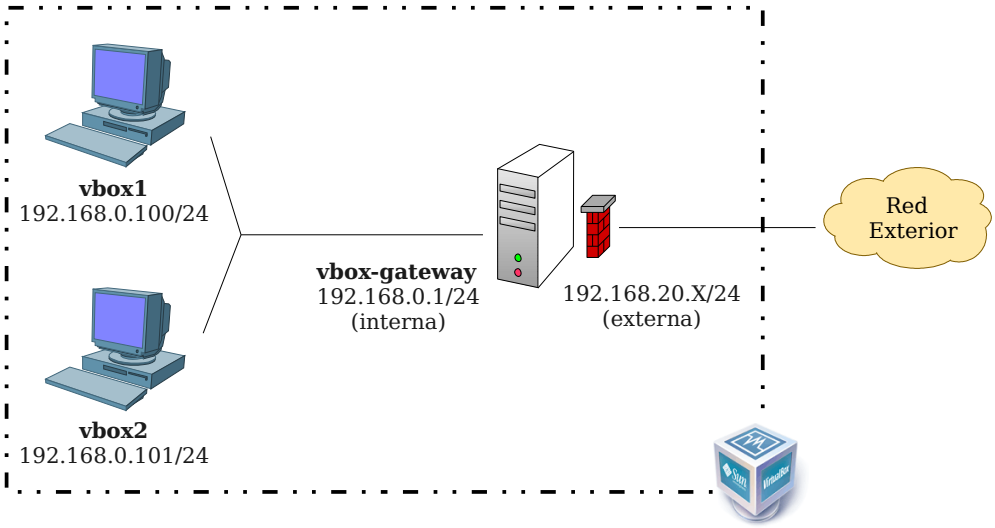

Figura 3. Esquema de configuración de la red virtual.

Para conseguir la configuración deseada hay que realizar modificaciones en la configuración de las máquinas en VirtualBox.

Tarea 3.- Asegúrese que las máquinas están apagadas y edite la configuración de la máquina *vbox1* y *vbox2*. En la sección Red de la configuración de la máquina establezca el Adaptador 1 como conectado a Red Interna (ver figura 4).

T3.1.- Para la máquina que hace de puerta de enlace se establecerán dos adaptadores de red. Edite la configuración de esta máquina y establezca el Adaptador 1 a la Red Interna y el Adaptador 2 al modo adaptador puente. Con ésto se conecta al adaptador físico de la máquina.

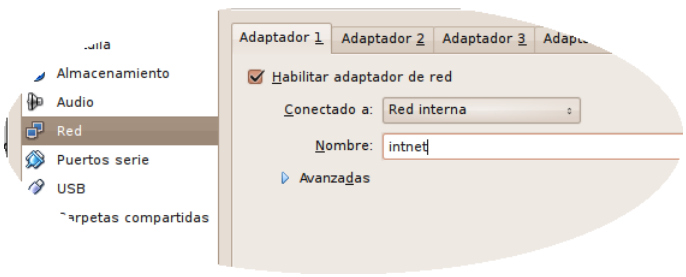

Figura 4. Configuración de adaptadores de red en VirtualBox.

2.2. Exportación de las máquinas virtualizadas

Antes de proceder a la preparación del entorno de trabajo se probará la exportación de la máquinas virtuales. Este proceso se debe realizar siempre tras la finalización de la sesión de laboratorio. Bastará con exportar la máquina que hace de gateway.

Tarea 4.- Desde el menú de VirtualBox seleccione el menú Archivo → Exportar servicio virtualizado y seleccione la máquina que hace de gateway o las tres máquinas simultáneamente. Exporte directamente en la memoria USB y mida el tiempo y espacio ocupado por la imagen.

3. Instalación y uso básico de Ubuntu Linux

La distribución Linux que usaremos en este curso es Ubuntu GNU/Linux. Aunque se ha preparado una imagen preinstalada para la realización de las sesiones de laboratorio, si se desea puede realizar la instalación completa en las máquinas virtuales. Para realizar la instalación considere que Ubuntu dispone de tres tipos de discos de instalación:

- Desktop (escritorio): Instalación del sistema a través de un escritorio “virtual”. Con esta opción se instalan programas de uso doméstico.
- Alternate: Similar al anterior salvo que su programa de instalación no es gráfico (requiere menos recursos) y proporciona opciones avanzadas de instalación.
- Server: Por defecto instala únicamente los componentes esenciales en un servidor (sin escritorio).

Las 3 opciones usan el mismo repositorio de paquetes y tras la instalación podrá elegirse cualquier software u opción de configuración independientemente de qué medio de instalación se empleó a priori.

Considere que el tamaño de una instalación de escritorio es de un tamaño bastante superior al preparado para esta asignatura.

3.1. Interfaz de comandos básica

Linux incluye una interfaz de comandos muy avanzada que facilita en gran medida las tareas de administración. Durante este curso se irán presentando los diferentes comandos necesarios para realizar la tareas solicitadas.

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)

Comando	Descripción
nano	Editor de textos
aptitude	Gestor de paquetes para instalación de software adicional.
exit	Cerrar el shell actual

Tabla 2. Comandos básicos en GNU-Linux

Tarea 5.- Entre con el usuario y la contraseña proporcionadas para probar los siguientes comandos:

T5.1.- Comando para solicitar ayuda sobre el comando *su*: `man su`.

T5.2.- Comando para convertirse en administrador del equipo: `sudo su` (se le solicitará de nuevo la contraseña).

T5.3.- Abandone la sesión utilizando el comando `exit`.

3.2. Configuración de la red

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 3. La configuración de red en *Ubuntu*, y en general de distribuciones basadas en *Debian*, reside en los siguientes ficheros de texto:

- /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 prioritario a las resolución de nombres mediante DNS.
- /etc/resolv.conf: Fichero con la lista de servidores de nombres disponibles.

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

Comando	Descripción
ifconfig	Configuración de interfaces
route	Manipulación de las rutas IP
ping	Envío de paquetes ICMP ECHO_REQUEST
traceroute / tracepath	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 6.- La primera tarea es cambiar el nombre de *Host* en cada una de las máquinas virtuales para evitar conflictos de nombres ya que, al haber sido clonadas todas tienen el mismo nombre. Hay que editar dos ficheros: `/etc/hostname` y `/etc/hosts`, debe utilizar la siguiente secuencia de comandos en cada una de las máquinas y establezca adecuadamente el nombre de Host en cada máquina.

```
sudo su
nano /etc/hostname
nano /etc/hosts
service hostname restart
hostname --fqd
```

Código 1. Comandos para el cambio de nombre de una máquina.

El siguiente paso es configurar la red en las diferentes máquinas, pero hay que distinguir entre las máquinas *vbox1* y *vbox2* frente a la que hace de gateway, la cual, posee dos interfaces de red.

Tarea 7.- Se configurará en primer lugar la red de *vbox1* y *vbox2* realizando estos pasos en cada una de las máquinas:

T7.1.- Convertirse en administrador (root) mediante el comando `sudo su`. Introduzca la clave adecuada.

T7.2.- Ejecute el comando `ifconfig -a` para ver los nombres de las interfaces y la configuración de la red. Supongamos que nos aparecen las interfaces *eth0*[1] y *lo*. La interfaz *lo* corresponde al bucle local (localhost/127.0.0.1) y *ethX*. Con un editor de texto, por ejemplo *nano*, edite el archivo /etc/network/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 192.168.0.100
    netmask 255.255.255.0
    gateway 192.168.0.1
```

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

T7.3.- Tras guardar los cambios en el fichero debe reiniciar la configuración de red utilizando el comando `/etc/init.d/networking restart` y comprobar la nueva configuración mediante el comando `ifconfig` y `route -v`.

T7.4.- Ejecute el comando `man interfaces` para consultar la documentación sobre este fichero.

T7.5.- Repita en la máquina *vbox2* estableciendo la dirección IP 192.168.0.101.

Tarea 8.- Cuando tenga configuradas dos máquinas ejecute el comando *ping* entre ellas para verificar el correcto funcionamiento de la red interna.

3.3. Configuración de la máquina GateWay

Esta máquina posee dos interfaces de red, al no disponer físicamente de conexiones en las interfaces se deben consultar y apuntar las direcciones MAC de los adaptadores para posteriormente saber cual es el que está conectado a la red interna y cual a la externa.

Tarea 9.- Utilice el comando `ifconfig -a` para identificar las interfaces de red interna y externa de su máquina gateway.

T9.1.- Una vez identificada establezca la configuración **únicamente para esta interfaz** repitiendo los pasos de T7.1.-, T7.2.- y T7.3.- usando la dirección IP 192.168.0.1.

1 Los nombres de intefaces de red el linux se numeran consecutivamente: eth0, eth1,eth2.

T9.2.- Pruebe con las tres máquinas virtuales funcionando el comando *ping* entre todas ellas.

Tarea 10.- La interfaz externa debe configurarla con una dirección IP proporcionada por el profesor para evitar conflictos con los compañeros de clase que estarán en la misma red. Esta red será 192.168.20.0/24

T10.1.- Solicite su dirección IP al profesor y edite de nuevo el fichero `/etc/network/interfaces` añadiendo la configuración de la nueva interfaz ethX como se indica:

```
# 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
```

Código 3. Configuración de la segunda interfaz de red en el gateway.

T10.2.- Cuidado en este momento, puede que tenga en el archivo de configuración dos gateway, para esta máquina su gateway es 192.168.20.1 por ello elimine de la configuración de la otra interfaz el gateway 192.168.0.1.

T10.3.- Reinicie la configuración de red utilizando el comando `/etc/init.d/networking restart` y comprobar la nueva configuración mediante el comando `ifconfig` y `route -v`.

T10.4.- Compruebe también con el comando *ping* que alcanza el gateway externo 192.168.20.1.

Tarea 11.- Para poder resolver los nombres de dominio es necesario configurar los DNS, para ello debe editar el archivo `/etc/resolv.conf` y establecer el siguiente contenido:

```
nameserver 150.214.130.15
```

Código 4. Configuración de servidores de nombres.

T11.1.- Compruebe la resolución de nombres y la conectividad en cada máquina utilizando el comando *ping* hacia www.dte.us.es.

T11.2.- Alternativamente puede configurar los DNS en el fichero /etc/network/interfaces añadiendo una línea con `dns-nameservers 150.214.130.15`.

Llegado a este punto se tiene la red interna y el gateway configurado pero podrá comprobar que aunque los equipos de la red interna alcanzan el gateway, estos ordenadores no consiguen salir al exterior a través del gateway. Para poner el gateway operativo será necesario configurar adecuadamente NAT en el gateway, lo cual se realizará en la siguiente sesión de laboratorio.

Tarea 12.- Visualice la ruta de los paquetes en su máquina gateway hasta los DNS públicos de Google mediante el comando `tracepath 8.8.8.8`.

T12.1.- Repita el proceso desde las máquinas internas de red para comprobar que su gateway no deja pasar la conexión.

4. 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 mas 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 Ubuntu podemos encontrar: Versiones soportadas anteriormente, Versión actual y Versión de desarrollo .

También para cada versión de la distribución suele tener varios componentes por motivos diversos. Por ejemplo, en Ubuntu:

- main: sección principal, libre y con soporte oficial.
- restricted: software necesario no libre.
- universe: software libre adicional, no soportado oficialmente.
- multiverse: software no soportado oficialmente con posibles problemas de distribución.

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 comando, los comandos básicos para gestión del software de Ubuntu son las mostradas en la tabla 4, de los cuales, se recomienda centrarse en *aptitude*.

Comando	**Descripción**
apt-get	Herramienta completa de gestión de paquetes APT
apt-cache	Busqueda de paquetes
aptitude	Interfaz amigable basada en Ncurses
apt-file	Busqueda de un fichero en el repositorio de paquetes
dpkg	Gestor de paquetes Debian

Tabla 4. Comandos básicos de gestión de paquetes.

4.1. Instalación de escritorio y pasos adicionales

En esta última sección 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 200MBytes, lo cual aumenta el tiempo y tamaño de exportación.

Existen multitud de escritorios disponibles en las diferentes distribuciones de GNU-Linux, los de menor peso y que consumen pocos recursos son *lx* y *xfce*. Pero, no basta con instalar los escritorios, se necesita un entorno gráfico capaz de manejar los controladores de vídeo siendo el más usado en Linux XORG.

Si instala XORG completo se instalarán todos los drivers para diferentes tarjetas gráficas, para ahorrar espacio instalaremos solamente los drivers para la tarjeta gráfica de Virtualbox. Por ello los paquetes a instalar son: v*irtualbox-ose-guest-x11* y *xfce4-session*. El sistema de dependencias de Debian detectará automáticamente los paquetes adicionales necesarios para que estos dos funcionen y los instalará y configurará automáticamente.

Tarea 13.- Antes de continuar se tomará una instantánea de la máquina gateway para poder restaurarla en caso de no gustar u ocupar demasiado espacio los resultados obtenidos. Desde el menú Máquina → Tomar instantánea establezca un nombre para la nueva instantánea

Tarea 14.- Como administrador en la máquina ejecute el comando `aptitude`. Aparecerá una aplicación con menús que deberá manejar con el teclado.

T14.1.- Usando la tecla F10 acceda a los menús y utilice la opción de menú Acciones → Actualizar la lista de paquetes. Espere a que termine la operación

T14.2.- De nuevo acceda al menú Buscar → Buscar y escriba literalmente "`virtualbox.*x11`" incluyendo el punto y el asterisco en el lugar adecuado. Usando la tecla "+" se seleccionará este paquete y todos los necesarios para que funcione, unos 200MB.

T14.3.- Repita el proceso para el paquete *xfce4-session* seleccionándolo para su instalación.

T14.4.- Acceda al menú Acciones → Instalar/eliminar paquetes y espere a que termine el proceso.

T14.5.- Abandone *aptitude* usando la tecla 'q'.

Tarea 15.- Una vez terminada la instalación, ejecute el comando `startx` y aparecerá un escritorio gráfico llamado *Xfce*.

4.2. Tareas adicionales y cuestiones

Tarea 16.- Antes de terminar la sesión de laboratorio exporte al menos la máquina que hace de gateway.

Tarea 17.- En la Tarea 13.- tomó una instantánea de la máquina, pruebe a restaurarla y ver como tanto la máquina como el tamaño del disco vuelven al estado original, es decir, antes de la instalación del escritorio gráfico. Los ficheros están en la ruta *VirtualBox* de su cuenta de usuario.

Durante la sesión de laboratorio pueden surgir las siguientes cuestiones:

Q1: ¿Por qué se renombran las interfaces al cambiar la dirección MAC y clonar la máquina? Debido a

ultradev, mirar fichero `/etc/udev/rules.d/70-persistent-net.rules` e interntar solucionarlo.

Q2: Si el comando `/etc/init.d/networking restart` falla continuamente hay que desconfigurar la red totalmente con estos comandos: `ifdown -a`, después `ifconfig ethX 0.0.0.0` en todas las interfaces y borrar la ruta por defecto con `route del default gw`.

Q3: En caso de obtener un error extraño con *tracepath* reinicie la máquina, puede que las tablas de rutas del núcleo estén mal configuradas debido a varios intentos fallidos de configuración de red.

Q4: Si se plantea la instalación la instalación de un servidor DHPC en su gateway para la red interna hay ser cuidadoso. Cuidado con esto, el gateway tiene dos interfaces de red, solo se deben atender peticiones en la interfaz de la red interna, ya que en la red externa, ya hay un servidor DHCP perteneciente encargado de suministrar IPs a los equipos de los laboratorios.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 3
Open VPN

Enunciados de Prácticas de Laboratorio
Tecnologías Avanzadas de la Información

1. Introducción y objetivos

OpenVPN es una solución VPN open source basada en SSL. Cubre un amplio rango de aplicaciones como: acceso remoto, unión de nodos en VPN, seguridad Wi-Fi, balanceo de carga, etc. Su principal ventaja frente a otras soluciones comerciales es su facilidad y reducido coste de implantación.

Esta solución opera en la capa 2 o 3 del modelo OSI uniendo mediante túneles todos los nodos distribuidos por la red. Requiere una instalación tanto en el cliente como en el servidor y es compatible con Linux, Windows 2000/XP/Vista/7, OpenBSD, FreeBSD, NetBSD, Mac OS X, y Solaris.

En esta sesión de laboratorio el alumno debe poner operativa una VPN en el entorno virtual de desarrollo. Se desarrolla de manera guiada en las primeras secciones la instalación y la puesta en marcha básica de la VPN. 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 5 donde se indica la máquina donde debe instalarlo.

Nombre del paquete	Descripción	Ubicación
xca	Utilidad gráfica para gestión de certificados digitales	Máquina local o máquina gateway (requiere entorno gráfico)
openvpn	OpenVPN	Todas la máquinas
mc	Administrador de archivos en para consola de texto	Todas las máquinas

Nombre del paquete	Descripción	Ubicación
openssh-server	Servidor ssh	Todas las máquinas

Tabla 5. Programas necesarios para la realización de la sesión de laboratorio.

2. Instalación de OpenVPN

La instalación de OpenVPN se realiza de manera similar a la de cualquier aplicación en Ubuntu/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 por ejemplo. La puesta en funcionamiento de OpenVPN requiere una configuración donde cada máquina puede tener diferentes perfiles (servidor VPN o cliente VPN o ambos). Por ello el sistema de paquetes Debian incluye una configuración por defecto, pero se facilita la configuración con una serie de ejemplos de configuración y herramientas para la gestión de certificados SSL.

Para realizar la sesión de laboratorio con mayor comodidad se recomienda la instalación de los paquetes adicionales de la tabla 5:

Tarea 18.- Debe realizar las siguientes instalaciones de paquetes para llevar a cabo todas las posteriores tareas:

T18.1.- Compruebe que tiene instalado el servidor *ssh* en todas las máquinas virtuales para poder realizar transferencias de ficheros entre máquinas. El paquete a instalar es *openssh-server*.

T18.2.- Instale el paquete *mc* es un administrador de archivos que funciona en la consola que, además, permite conexiones remotas para copiar archivos entre diferentes máquinas de la red.

T18.3.- Instale el paquete *xca*, es una utilidad de gestión de certificados, aunque no es necesario facilitará en gran medida el desarrollo de la sesión de laboratorio.

A continuación se instalará OpenVPN, tras la instalación los archivos de configuración deben ubicarse en el directorio /etc/openvpn pero por defecto, no trae establecida ninguna configuración. El directorio /usr/share/doc/openvpn trae documentación y ejemplos para la puesta en funcionamiento. En este mismo directorio trae un conjunto de scripts para generar las claves RSA y certificados, pero para mejorar la comprensión del proceso se recomienda el uso de XCA.

Tarea 19.- Instale el paquete *openvpn* en la máquina gateway, como se ha comentado, no trae ninguna configuración predeterminada, por lo que no activará ningún servicio en la máquina.

T19.1.- 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 sample-config-files con ejemplos para utilizar en la configuración tanto del cliente como el servidor.

T19.2.- 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:

T19.2.1.- 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.

T19.2.2.- La opción rápida (no recomendada) es utilizar las utilidades incluidas en el directorio easy-rsa. Para ello debe seguir los pasos indicados en el howto de OpenVPN en la siguiente

dirección http://openvpn.net/index.php/open-source/documentation/howto.html#pki. Si elige esta opción obvie la Tarea 20.- y la Tarea 21.-

2.1. 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 utilizará una autoridad de certificación que firmará todos los certificados y con posibilidad de incluir una lista de revocación. Todos estos componentes forman una PKI utilizada para operar en la VPN.

Toda la fortaleza de la seguridad de 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 el 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 incluyendo la clave pública. La autoridad de certificación firma el certificado digital del servidor.
- Cada cliente de la VPN genera su clave pública, su clave privada y un certificado digital que incluye su clave pública. De manera similar la autoridad de certificación firma el certificado de cada cliente.
- El administrador publica para todos los clientes/servidores la clave pública de la autoridad en un certificado, para que así puedan validar las firmas digitales.

En la tabla 6 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 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**, solo 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 cliente correspondiente con los permisos adecuados para evitar su copia	**Sí**, sólo accesible para el cliente

Contenido	Ubicación	Secreto
Certificado firmado con la clave pública de cada cliente	Cada certificado sólo en el cliente correspondiente	**No**

Tabla 6. 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.

Tarea 20.- Instale el paquete *xca* desde el gestor de paquetes e inicie el programa. Se utilizará este programa para crear los certificados de la siguiente forma:

T20.1.- Inicie el programa *xca* y cree una nueva base de datos para almacenar sus certificados.

T20.2.- 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 Certificates y pulse el botón New certificate. Utilice la figura 5 para configurar correctamente las dos primeras pestañas, el resto de pestañas manténgalas con los valores predeterminados.

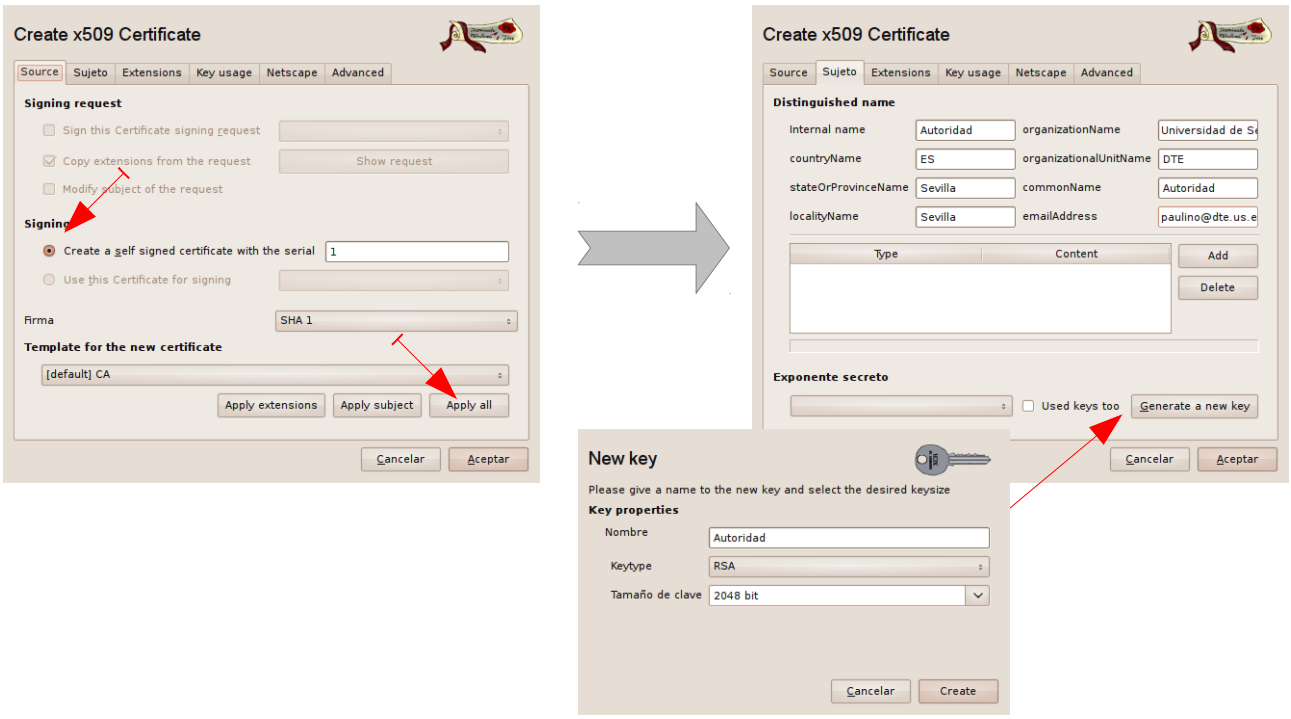

Figura 5. Generación de la autoridad de certificación.

T20.3.- El siguiente paso es crear un certificado para el servidor firmado por la autoridad certificadora. Para ello, cree un nuevo certificado y configure las diferentes opciones según se indica en la figura 6, dejando las opciones no mostradas a su valor predeterminado.

T20.4.- 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 y con la configuración mostrada en la figura 7.

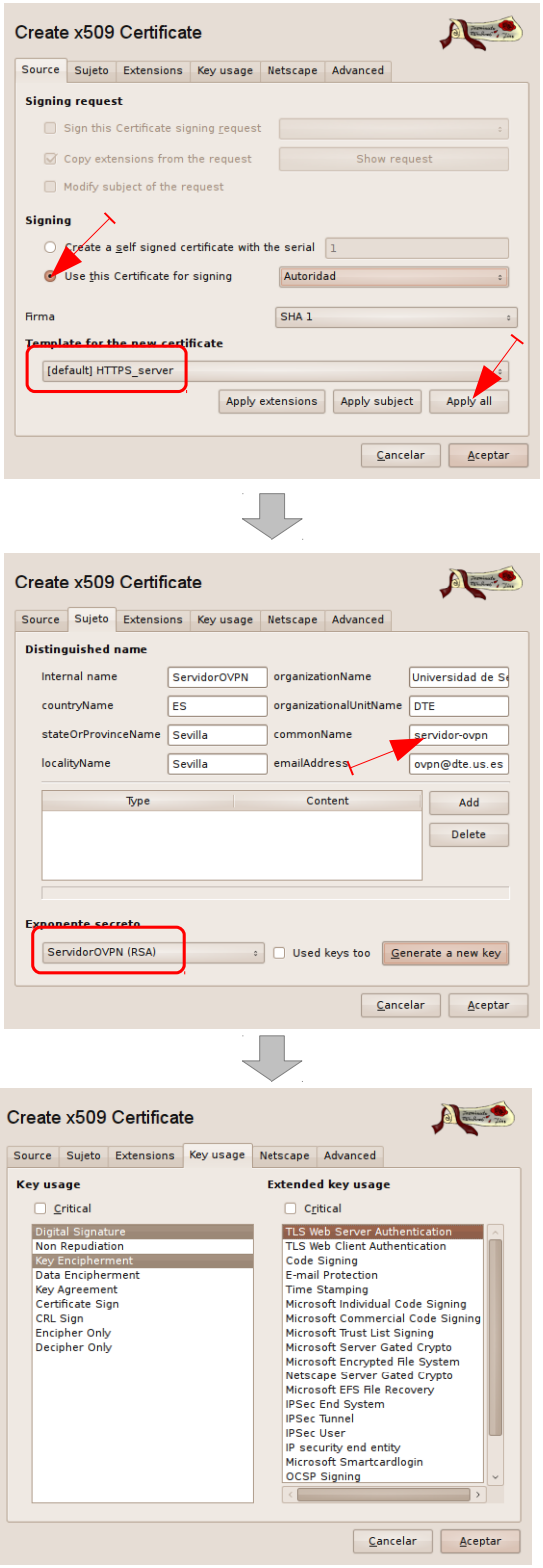

Figura 6. Generación del certificado del servidor.

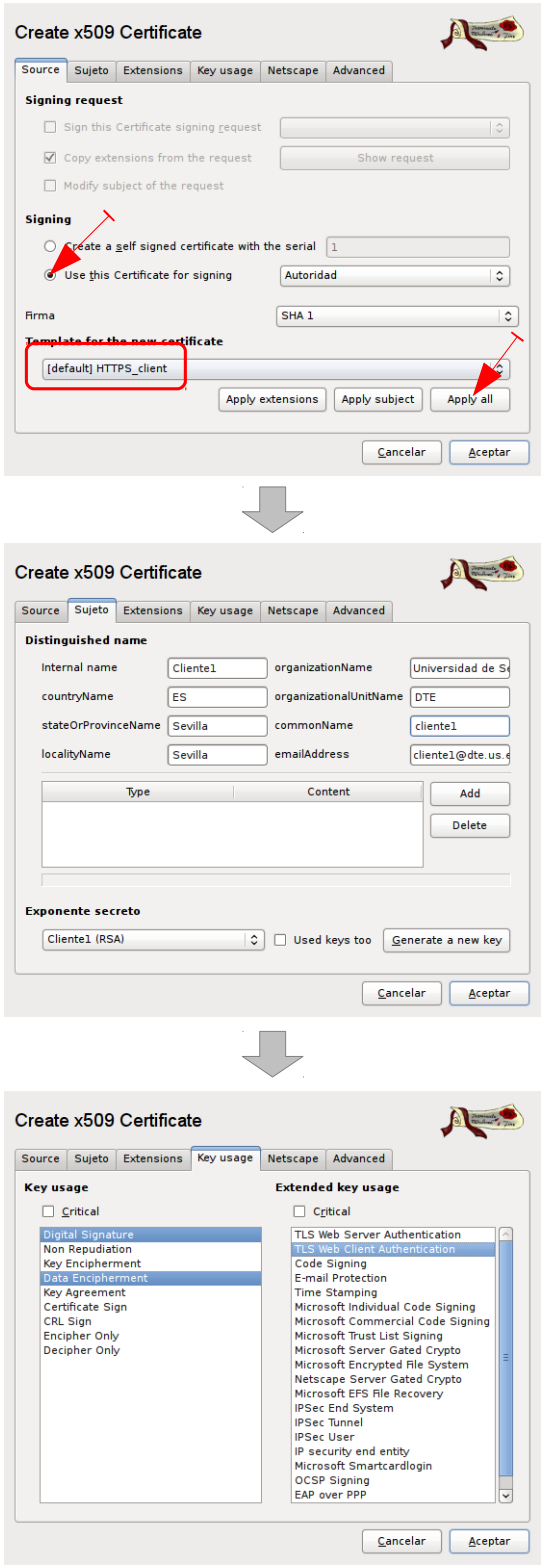

Figura 7. Generación del certificado del cliente.

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

Tarea 21.- Se procederá a la exportación de los certificados y de las claves privadas de la siguiente forma:

T21.1.- Se exportarán las claves privadas del servidor y del cliente. No debe exportar la clave privada de la autoridad, esta 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 las claves del cliente, el servidor y use el botón Export. Use las opciones predeterminadas en la exportación de los ficheros, se generarán ficheros con extensión *.pem*.

T21.2.- Ahora se procederá a la exportación de los certificados desde la pestaña Certificates. Seleccione todos los certificados, incluido la autoridad, y expórtelos con las opciones por defecto, obtendrá diferentes ficheros con extensión *.ctr*. Estos ficheros contienen las claves públicas.

2.2. 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 8 para tener una visión global de cómo deben quedar los certificados y claves en cada una de los equipos virtuales.

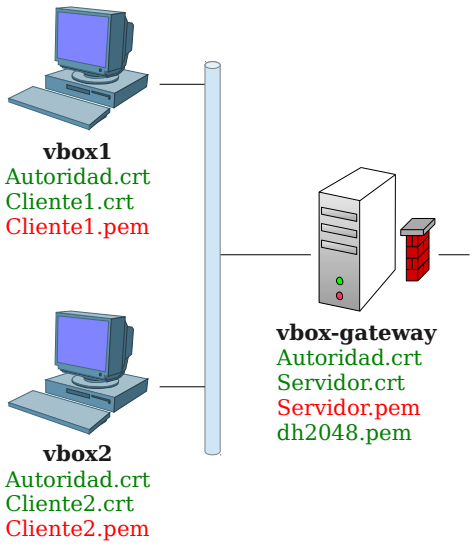

Figura 8. Ubicación de certificados y claves en la VPN..

Tarea 22.- Para comenzar la configuración del servidor asegúrese de tener instalado el paquete *openvpn*.

T22.1.- La configuración del servidor requiere un fichero adicional llamado parámetros de *Diffie-Hellman* utilizados para el intercambio seguro de claves privados. Desde XCA genere este fichero usando el menú File → Generate DH parameter. Obtendrá un fichero necesario sólo en el servidor.

T22.2.- En el servidor debe copiar 4 ficheros en la ubicación `/etc/openvpn`, deberá transferirlos por *ssh*, puede utilizar el comando *scp* o el programa *mc*. Los ficheros son los indicados en la tabla 7, los nombres pueden variar según como los tenga guardados de la tarea anterior.

Fichero	Contenido
Autoridad.crt	Certificado de la autoridad certificadora con su clave pública
dh2048.pem	Parámetros Diffie-Hellman generados en T22.1.-
Servidor.crt	Certificado del servidor con la clave pública firmado por la autoridad de certificación
Servidor.pem	Clave privada del servidor

Tabla 7. Ficheros necesarios en el servidor OpenVPN.

T22.3.- Copie el fichero `/usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz` que contiene una configuración de ejemplo a `/etc/openvpn` y descomprímalo con el programa `gunzip` desde la interfaz de comandos.

T22.4.- 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.
# OpenVPN can also use a PKCS #12 formatted key file
# (see "pkcs12" directive in man page).
ca Autoridad.crt
cert Servidor.crt
key Servidor.pem  # This file should be kept secret
```

Código 5. Fichero de configuración para el servidor OpenVPN.

T22.5.- Inicie el servicio OpenVPN mediante el comando `/etc/init.d/openvpn start`. Si todo ha ido bien, con el comando `ifconfig` aparecerá una nueva interfaz llamada `tun0` con una IP asignada, esta es la interna de la VPN IP perteneciente al servidor. Si ha tenido problemas revise la Q3 en el último capítulo de este documento.

Para continuar con la configuración hay que integrar los clientes en la VPN. Cada uno de los clientes necesitará tres ficheros para poder unirse a la VPN:

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

Tarea 23.- Para configurar un cliente siga los siguientes pasos:

T23.1.- 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`.

T23.2.- Transfiera por *ssh* o con *mc* los tres ficheros necesarios a la ubicación `/etc/openvpn` del cliente.

T23.3.- Edite el fichero `/etc/openvpn/client.conf` estableciendo los valores correctos para los certificados y claves tal y como se mostró en el código 5. Considere que ahora los dos últimos ficheros tienen nombre diferente.

T23.4.- 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 192.168.0.1 1194`.

T23.5.- Reinicie el servicio mediante `/etc/init.d/openvpn restart` y ejecute `ifconfig` para comprobar si se ha conectado la interfaz `tun0` y tiene una IP asignada.

T23.6.- Utilice tanto en el servidor como en el cliente el comando `less /var/log/syslog` para ver el resultado de la negociación entre el cliente y el servidor de la VPN.

Si tiene problemas de conexión por parte del cliente lea la última sección de este documento donde se comentan algunos problemas típicos que surgen durante el proceso de configuración.

Tarea 24.- Añada la segunda máquina de la red interna a la VPN generando nuevos certificados y repitiendo los pasos de la tarea anterior.

Tarea 25.- En el servidor VPN visualice el contenido de los ficheros del directorio /etc/openvpn llamados openvpn-status.log y ipptxt. Interpretando correctamente el contenido realice lo siguiente:

T25.1.- Ejecute el comando *ping* desde el servidor a los clientes a través de la VPN

T25.2.- Ejecute el comando *ping* desde un cliente a otro cliente a través de la VPN

T25.3.- En el paso anterior los paquetes no llegaron entre los diferentes equipos de la VPN. Edite el fichero de configuración del servidor VPN y busque una directiva que habilite la visibilidad entre los diferentes clientes. Tras los cambios debe reiniciar el servicio OpenVPN tanto en el servidor como en todos los clientes.

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.

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

T26.1.- Ahora seleccione el certificado de la autoridad de certificación y vuelva a desplegar el menú con el botón derecho. Aparecerá un submenu CA → Generate CRL.

T26.2.- 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 27.- El último paso es cambiar la configuración del servidor OpenVPN.

T27.1.- Copie el fichero CRL llamado *AutoridadCRL.pem* en el directorio /etc/openvpn del servidor.

T27.2.- Edite la configuración del servidor (fichero /etc/openvpn/server.conf) y añada la directiva indicada en el código 6 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 6. Configuración del listado CRL en OpenVPN.

T27.3.- Reinicie el servicio OpenVPN del servidor con `/etc/init.d/openvpn restart` y utilice el comando `tail -f /var/log/syslog` para observar el comportamiento del servidor. Debe reiniciar los clientes ya que tardan cierto tiempo en reconectar al servidor.

T27.4.- ¿Observa el aviso en el servidor sobre el cliente con el certificado revocado? ¿Que ocurriría si se revoca el certificado del servidor?

4. Modos de funcionamiento de OpenVPN

Una vez conseguido un funcionamiento básico de OpenVPN, se propone probar diversas configuraciones y modos de funcionamiento admitidos. Si lee detenidamente los comentarios del fichero de configuración del servidor, observará multitud de ejemplos y directivas comentadas que alteran el funcionamiento predeterminado.

Las siguientes tareas debe resolverlas cambiando los ficheros configuración del servidor y los clientes según los ejemplos indicados en los propios ficheros de configuración.

Tarea 28.- ¿Como está funcionando su VPN usando UDP o TCP? Averígüelo y cambie el modo de funcionamiento. Use el comando `netstat -ltnp` para ver el puerto donde está el servidor.

Tarea 29.- Establezca una IP fija para cada cliente, para ello mire el ejemplo indicado donde se utilizan ficheros ubicados en un directorio llamado *ccd*.

T29.1.- Edite el fichero hosts en todos los equipos dando un nombre distintivo a cada equipo de la VPN, por ejemplo: *vpn-gateway*, *vpn-vm1* y *vpn-vm2*.

T29.2.- Ejecute el comando *ssh* utilizando estos nombres de hosts para ver si la VPN opera correctamente.

Tarea 30.- Configure un cliente Windows para que se conecte a su VPN, para ello descargue el cliente para Windows existente en la página oficial de OpenVPN.

Tarea 31.- Ahora el objetivo es conectar una de las máquinas virtuales a la VPN de otro compañero a través de una solicitud, es decir, su compañero no debe generar su clave secreta, solo debe operar con la pública.

T31.1.- Debe generar una clave privada y certificado sin firmar en su XCA.

T31.2.- Exporte el certificado, pero mantenga su clave privada oculta. Pídale a un compañero que importe su certificado y lo firme con la autoridad de su VPN.

T31.3.- Con el certificado firmado, su clave privada y el certificado de la autoridad de su compañero configure una máquina para que se conecte a la VPN externa de su compañero.

Tarea 32.- Consiga usar un único certificado para todos los clientes, por defecto no funciona. Configure correctamente el servidor para que esto sea posible.

Los dos siguientes ejercicios presentan cierta dificultad y requieren tiempo, se consideran opcionales,

puede realizarlos si está interesado en ampliar sus conocimientos sobre OpenVPN.

Tarea 33.- Un método habitual para dotar de seguridad a una red de área local es encaminar todo el tráfico IP o parte, y tanto local como externo a través de la VPN.

T33.1.- Intente configurar los equipos cambiando las rutas para que use como gateway por defecto el servidor VPN.

T33.2.- Configure el servidor para que realice NAT a los clientes de la VPN.

Tarea 34.- Se propone un ejercicio más complicado, su servidor debe operar como cliente y servidor simultáneamente. Será servidor VPN en una red diferente a la red a la que se conecte.

T34.1.- Para evitar problemas con las direcciones IP debe cambiar el número de red por defecto sobre la que opera su servidor OpenVPN.

T34.2.- Necesitará una certificado firmado por la autoridad de otra VPN y crear en su servidor una configuración adicional como cliente. En esta configuración considere usar un dispositivo túnel diferente de `tun0`, este ya está usado para operar como servidor.

T34.3.- Una vez conseguida la existencia de dos VPNs en su servidor, la tarea se complica si intenta activar el reenvío de paquetes de una red VPN a otra. ¿Es posible conseguir que un equipo de una VPN se conecte al de la otra VPN usando el servidor como puente para reenviar los paquetes?

5. Recomendaciones y FAQ

Q1: Al tener poca memoria puede que el sistema operativo mate procesos, es recomendable añadir un fichero del disco como memoria de paginación (swap). Debe ejecutar las secuencia de comandos siguiente:

```
dd if=/dev/zero of=/swap.fs bs=1M count=256
mkswap /swap.fs
```

Ahora para añadirlo al sistema de manera permanente debe editar el fichero `/etc/fstab` añadiendo una línea con la siguiente sintaxis

```
/swap.fs   none    swap    sw     0      0
```

Para activar y comprobar si tiene memoria de paginación mediante los comandos

```
swapon -a
free -m
```

Q2: Para comprobar si un certificado está firmado correctamente ejecute el siguiente comando.

```
openssl verify -CAfile ca.crt client1.crt
```

Q3: En caso de no iniciar correctamente el servicio VPN la interfaz `tun` no parecerá, debe comprobar lo ocurrido en la bitácora del sistema `/var/log/syslog`. Si no hay suficiente información puede indicar a *OpenVPN* que muestre más información cambiando en el fichero el configuración el nivel de la variable `ver`, se recomienda no pasar de 4.

Q4: Si en el fichero `/var/syslog` aparecen errores del tipo `TLS Error` el principal motivo puede ser una generación de certificados incorrecta. Compruebe que los certificados contienen los campos siguientes

desde el botón Show Details de XCA:

- Los certificados de los clientes deben tener el campo keyUsage=digitalSignature, keyEncipherment
- El certificado del servidor debe tener el campo Key Usage = Digital Signature, Key Agreement

Otra solución para evitar este problema comentar en el archivo de configuración la directiva ns-cert-type.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 4
Servicios de red

Enunciados de Prácticas de Laboratorio
Tecnologías Avanzadas de la Información

1. Introducción y objetivos

La duración estimada de esta sesión de laboratorio es de **2 horas**. El propósito general de esta sesión de laboratorio consiste en realizar un despliegue de diferentes servicios entre los equipos de la red virtual. Este despliegue será necesario para realizar el laboratorio posterior dedicado al *Control de Tráfico* y *Calidad de Servicio*.

Concretamente se instalarán un servidor WEB y un servidor FTP. Los servicios instalados serán públicos a través del la máquina que hace de puerta de enlace. Para conseguirlo se propone configurar adecuadamente *netfilter* de Linux y realizar reenvío de paquetes a la máquina que sirve cada uno de los servicios.

Nombre del paquete	Descripción	Ubicación
firefox	Navegador Web	Máquina Gateway
filezilla	Programa para transferencias FTP/FTS/SFTP	Máquina anfitrión de Virtualbox

Tabla 8. Paquetes recomendados para la realización del laboratorio.

2. Instalación de servicios

Se instalarán dos servicios, un servidor WEB en la máquina *vbox1* y un servidor FTP en la máquina *vbox2*. En la distribución de Linux utilizada existen multitud de alternativas para la instalación de ambos servidores de hecho, en la tabla 9 se indican las soluciones más habituales utilizadas.

Tipo de Servidor	Nombre del paquete
WEB	lighttpd
	apache2
	cherokee
FTP	proftpd
	vsftpd
	pure-ftpd
	wu-ftpd

Tabla 9. Servidores de uso común en Linux.

Independientemente de las opciones que se escojan, la configuración de los servicios se pueden consultar en la documentación disponible para cada uno de los programas. Se deben escoger dos de ellos a libre elección del alumno y en caso de encontrar alguna dificultad se puede cambiar a alguna otra de las alternativas indicadas en la tabla.

Respecto al servidor WEB se deben ofrecer 2 servicios: un en puerto estándar HTTP (puerto 80) para páginas sin conexión segura, y otro para HTTPS (puerto 443) donde se servirán páginas seguras usando SSL. Para la puesta en marcha del servidor realice los siguientes pasos:

Tarea 35.- Instale un servidor WEB en la máquina *vbox1*. Compruebe tras la instalación navegando desde la máquina que hace de puerta de enlace si aparece una página WEB mediante la dirección http://192.168.0.100

T35.1.- El siguiente paso es activar SSL en el puerto 443. Genere un certificado digital con la clave pública y privada utilizando el programa XCA para utilizarlo en el servidor WEB.

T35.2.- Configure el servidor WEB para que opere en el puerto 443 con los certificados generados anteriormente. Para ello consulte la documentación disponible del servidor elegido.

Para el servidor FTP cualquiera de las alternativas de la tabla opera de forma similar, sólo cambian los ficheros de configuración. Tras la instalación del servidor FTP podrá comprobar que el servidor opera aceptando conexiones con la identificación de los usuarios existentes en el sistema, considere que esta configuración por predeterminada es un potencial agujero de seguridad por diversos motivos:

- La conexión FTP no está cifrada y 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 36.- Escoja alguna de las opciones de la tabla para instalar un servidor FTP en la máquina *vbox2*.

T36.1.- Añada varios usuarios al sistema mediante el comando `adduser`. Compruebe si pueden

conectarse realizando pruebas desde la máquina que hace de puerta de enlace. Puede utilizar desde la línea de comandos el comando `ftp` o instalar en esta máquina *filezilla*.

T36.2.- Usando la documentación del servidor FTP que ha instalado intente activar el soporte STARTTLS. Configúrelo generando nuevos certificados con XCA para intentar cifrar las conexiones FTP y evitar problemas de seguridad.

3. Configuración de la puerta de enlace

Tras el despliegue de los servicios en la red interna se debe configurar el firewall de la puerta de enlace de forma que redirija servicios públicos a los equipos internos que gestionan cada servicio. En la figura 9 se muestra esquemáticamente como debe configurarse el firewall.

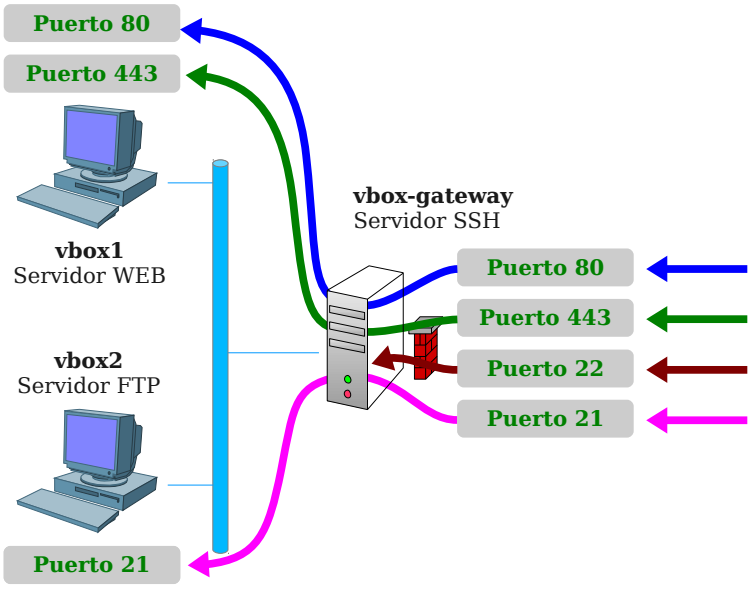

Figura 9. Esquema de configuración del firewall.

Además de configurar el firewall correctamente, se debe considerar un procedimiento para que las reglas del firewall se carguen automáticamente tras el reinicio de la máquina. Para conseguirlo basta con saber que el proceso de arranque de Ubuntu ejecuta automáticamente el fichero /etc/rc.local. Este fichero es un fichero de texto donde se pueden colocar los comandos adicionales que se desean ejecutar. Es muy importante que este fichero devuelva el valor '0' tras su ejecución, lo cual, está indicado enfáticamente en el propio fichero. Deberá utilizarlo de manera adecuada para automatizar la configuración del firewall.

Trabajando en la máquina que hace de puerta de enlace realice lo siguiente:

Tarea 37.- Como administrador cree un nuevo directorio en la cuenta de root de la forma /root/bin, aquí se ubicarán todos los *scripts* generados durante todas las sesiones de laboratorio.

T37.1.- En este directorio cree un nuevo fichero llamado firewall.sh y añada permiso de ejecución al mismo mediante el comando `chmod +x firewall.sh`.

T37.2.- Edite el fichero /etc/rc.local y antes de la sentencia `exit 0` añada una línea invocando al script que contiene el firewall, debe quedar `/root/bin/firewall.sh`

Tarea 38.- Trabajando en el fichero /root/bin/firewall.sh añada reglas de netfilter para que opere de la forma indicada en la figura 9. Las reglas serán del tipo -j DNAT y son para conseguir lo siguiente:

T38.1.- Todo el tráfico que se reciba por la interfaz externa (192.168.20.X) en el puerto 80 debe redirigirse al puerto 80 de la máquina *vbox1*.

T38.2.- Todo el tráfico que se reciba por la interfaz externa (192.168.20.X) en el puerto 443 debe redirigirse al puerto 443 de la máquina *vbox1.*

T38.3.- Todo el tráfico que se reciba por la interfaz externa (192.168.20.X) en el puerto 21 debe redirigirse al puerto 21 de la máquina *vbox2.*

Tarea 39.- Compruebe con un navegador, desde la máquina anfitrión de Virtualbox, o desde otro ordenador del aula la redirección de los servicios, para ello:

T39.1.- Navegue a la dirección http://192.168.20.X

T39.2.- Navegue a la dirección https://192.168.20.X

T39.3.- Use un cliente FTP para conectar mediante ftp://192.168.20.X, pruebe los modos de operación activo y pasivo ¿funcionan ambos?

T39.4.- Para poner operativo el modo activo FTP pruebe cargar los siguientes módulos mediante los comandos indicados:

```
modprobe ip_conntrack
modprobe ip_conntrack_ftp
modprobe ip_nat_ftp
```

T39.5.- Para automatizar la carga de estos módulos tras el reinicio del equipo tiene dos opciones: (1) Añadir los comandos en el fichero del firewall (2) Añadir los sólo los nombres de los módulos a cargar en el fichero de texto /etc/modules.

4. Recomendaciones y Cuestiones

Q1: La velocidad de instalación de paquetes depende de la fuente de los repositorios, se recomienda utilizar ubuntu.cica.es. Para ello edite el fichero /etc/apt/sources.list y cambien las URLs. Alternativamente descargue el fichero sources.list desde la página de material de la asignatura y cópielo a la ubicación /etc/apt/.

Q2: Aunque ya se ha hecho anteriormente se recuerda que para activar el reenvío de paquetes de manera permanente se puede cambiar en el fichero /etc/sysctl.conf quitando el comentario de la línea: `net.ipv4.ip_forward=1`.

DEPARTAMENTO DE TECNOLOGÍA ELECTRÓNICA
ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Laboratorio 5
QoS y Control de tráfico

Enunciados de Prácticas de Laboratorio
Tecnologías Avanzadas de la Información

1. Introducción y objetivos

Linux implementa 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. El tiempo estimado de realización de este laboratorio es de 4 horas.

Ante la multitud de posibilidades existentes en Linux, en este laboratorio se describirán algunas y probarán un conjunto reducido de ellas, concretamente aquellas 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 tanto de la configuración como de los resultados obtenidos al aplicar diferentes disciplinas. En un segundo bloque, se requiere la realizar una 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 de laboratorio se debe disponer del material de la tabla 10 y en la ubicación indicada.

	Descripción	**Ubicación**
filezilla	Programa para realizar transferencias FTP y SFTP	Máquina anfitrión
descargas.sh	Script para generar descargas simultáneas	Máquina anfitrión

Tabla 10. 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 http://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 cola de las 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, llamados filtros.

En la implementación realizada en Linux las disciplinas de colas que se desee usar se asocian en forma de árbol, forman una jerarquía de disciplinas cuyo nodo raíz es una interfaz de red. Posteriormente a ciertos nodos se adjuntan reglas para decidir cual es el nodo destino de cada paquete, estas reglas se denominan filtros. Algunas disciplinas de colas van más allá de ser un simple nodo de la jerarquía, incluyen subnodos llamadas clases hijas donde se pueden encolan los paquetes.

En la figura 10 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 forman parte del árbol: disciplinas, clases y filtros.

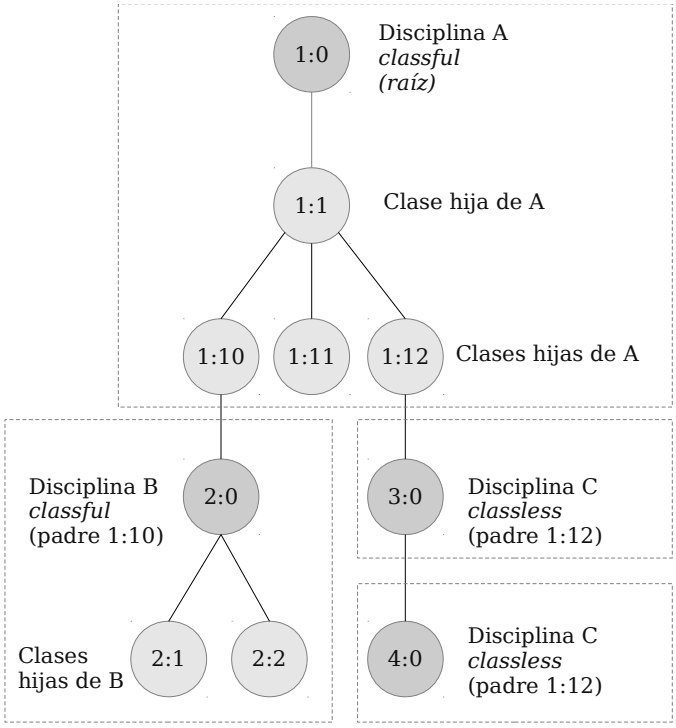

Figura 10. Jerarquía de ejemplo

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 una sólo disciplina se consigue esta configuración.

Pero no todas las disciplinas implementadas disponen de clases hijas distiguiéndose dos tipos de disciplinas de colas: 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 tienen 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.

Respecto a la clasificación de los paquetes, anteriormente se han presentado los filtros como reglas de clasificación. Para utilizar correctamente los filtros se debe conocer como el núcleo de Linux interactúa con el árbol jerárquico de disciplinas creado por el administrador del sistema. Siempre, el punto de entrada de los paquetes es la raíz de la jerarquía, así, cuando los paquetes llegan a una interfaz se les aplican los filtros asociados a la raíz de la jerarquía. Con estos filtros se decide en que clase de todo el árbol se encola el paquete. Aunque no se recomienda, también es posible tener filtros asociados a otros nodos internos de la jerarquía y con este tipo de configuración, cuando un nodo inferior recibe un paquete pueden de nuevo aplicarle más filtros asociados al nodo siendo de nuevo movido a otro hijo dentro de la jerarquía. Esta solución compleja sólo tiene sentido pasa su aplicación es situaciones donde existen multitud de filtros, el sistema de filtrado se optimiza distribuyendo filtros por el árbol ya que disminuye el tamaño de las listas de filtros en cada nodo.

Otro aspecto importante es conocer como el sistema de filtrado identifica los nodos inequívocamente para poder ubicar 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 explicita 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.

- SQF: *Stochastic Fairness Queueing*, es una disciplina llamada estocástica equitativa basada en l 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 sea clasificado por un filtro.

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 mostrarán los aspectos más relevantes de las diferentes disciplinas pero entrando sólo a detallar de 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, cada modo para manipular un elementos elemento diferente del sistema de control de tráfico: disciplinas, clases y filtros.

La sintaxis para configurar disciplinas y clases depende de cada disciplina usada, y se mostrarán diversos ejemplos en la descripción de cada disciplinas considerada de interés. No obstante, los filtros funcionan de manera similar para todas las disciplinas y se dedicará una sección adicional en este documento para explicarlos debido a su complejidad.

De forma general, se utilizará el comando *tc* según lo indicado en la tabla 11, y se debe considerar un aspecto adicional que son las unidades utilizadas con este comando, ya que pueden crear confusión. En la tabla 12 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** eth0 **root** tbf ...	Añadir disciplina TBF como raíz
	tc qdisc add eth0 **parent** 1:0 **handle** 10: sqf ...	Añadir disciplina SQF como hija de la disciplina 1:0
	tc qdisc del dev eth0 **root**	Eliminar todas las disciplinas de la interfaz eth0 (limpieza)
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 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 11. Modos de operación del comando tc.

Sintaxis	Tipo de unidad	Unidad	Equivalencia
kbps	Caudal	Kilobytes por segundo	1024bps
mbps	Caudal	Megabytes por segundo	1024 × 1024bps
kbit	Caudal / Tamaño	Kilobits por segundo / Kilobits	1024bit
mbit	Caudal / Tamaño	Megabits por segundo / Megabits	1024 × 1024bit
bps	Caudal	Bytes por segundo	
kb	Tamaño	Kilobytes	1024b
mb	Tamaño	Megabytes	1024 × 1024b
b	Tamaño	Bytes	
s, sec, secs	Tiempo/Tamaño	Segundos	
ms, msec, msecs	Tiempo/Tamaño	Milisegundos	10^{-3} sec
us, usec, usecs	Tiempo/Tamaño	Microsegundos	10^{-6} sec

Tabla 12. 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 los paquetes con filtros. 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 solo 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 mantienen estadísticas del funcionamiento de la cola y pueden consultarse fácilmente. Una disciplina de este tipo puede se 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, la diferencia está en que una disciplina *classful* puede contener tantos filtros añadidos como se desee para analizar los paquetes. En cambio PFIFO_FAST no admite filtros, se asignan paquetes a cada banda analizando únicamente el campo TOS, no se puede establecer otro tipo de filtro para realizar clasificación adicional.

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, maximizar fiabilidad y minimizar coste.

2.2.3 Filtro de 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, ethernet usa 64 bytes de cabeceras. 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 WFQ. 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 admiten filtros para clasificar los 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 que presenta mejoras significativas respecto a CQB, tanto a nivel de rendimiento como de 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, lo 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 11 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 menor 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 11:

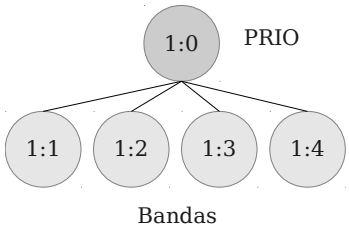

Figura 11. Disciplina PRIO con 4 bandas

```
# 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:
```

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 pueden 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 para 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 disponibles 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 con el sobrante, solo se permite a cada clase alcanzar un máximo también establecido en la configuración. Además de controlar el máximo/mínimo caudal en cada clasem 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 prestan 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 y es la clase predeterminada donde se encola el tráfico. Tras esto, se configuran todas las clases hijas que se deseen en forma de árbol, en cada uno de los hijos del árbol en donde se establecen los siguientes parámetros: caudal mínimo garantizado, caudal máximo alcanzable y prioridad en la adquisición de caudal disponible.

Para hace un uso adecuado de HTB se requiere conocer el algoritmo seguido cuando se decide de que clase del árbol de clases HTB se extrae un paquete para su envío. 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 los hijos siguiendo un proceso basado en prioridades. Así, cada clase hija tiene configurada una prioridad, y en igualdad de prioridades entre hijos, 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, alguna ya se presentado con anterioridad:

- Los nodos intermedios del árbol no pueden contener colas de paquetes, por ello, el destino de los filtros 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 12. En este ejemplo 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: Velocidad / Caudal garantizado.
- Ceil: Velocidad / Caudal máximo alcanzable.
- Prio: Prioridad en la obtención de caudal sobrante.

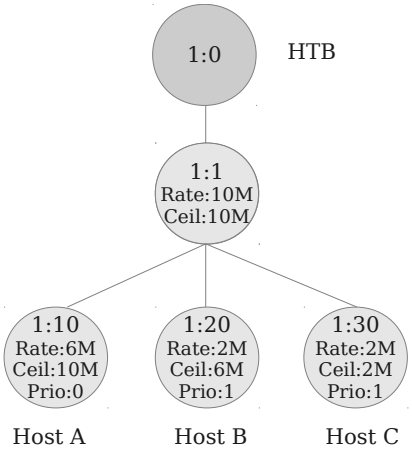

Figura 12. Jerarquía de ejemplo HTB

Con la configuración y parámetros mostrados en la figura 12 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 0 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 1 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 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 12, 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) = 8M* la padre clase 1:1 dispone de 2M 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 2M a clase 1:10. El resultado es que el equipo A usa un caudal de 8M y el equipo B sólo dispone de su caudal garantizado de 2M.
- 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 6M 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 6M y el equipo C 2M, ambos alcanzan el caudal *Ceil* (máximo) y a la clase padre le sobran 2M 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, significa la clase padre no puede 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 12:

```
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 0
tc class add dev eth0 parent 1:1 classid 1:20 htb rate 2mbit ceil  6mbit prio 1
tc class add dev eth0 parent 1:1 classid 1:30 htb rate 2mbit ceil  2mbit prio 1
```

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. En la especificación de la disciplina HTB sólo se puede indicar la clase por defecto pero no ningún otro parámetro relacionado con la conformación 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 conformado de tráfico (*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 12.
- 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 la clases HTB. Cuando 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 13 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 selección entre las clases hijas
rate [caudal]	Velocidad garantizada para este clase (y sus hijos)
ceil [caudal]	Velocidad máxima de envío para esta clase (y sus hijos)
burst [bytes]	Tamaño en bytes de la ráfaga que puede superar la velocidad máxima

Parámetro	Descripción
cburst [bytes]	Tamaño en bytes de la ráfaga que se pueden 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 13. Parámetros admitidos en las clases de la disciplina HTB.

2.4. Clasificación mediante filtros

La clasificación consiste en aplicar reglas a los paquetes para establecer el destino final en el árbol jerárquico de disciplinas de la interfaz. Estas reglas se llaman filtros y contienen una serie de condiciones que deben cumplir los paquetes.

Los filtros se adjuntan a determinados nodos de la jerarquía de disciplinas y forman una lista de filtros. Cada nodo puede contener más de una lista de filtros, cada lista se diferencia por una prioridad representada por un número natural. Así, cuando un filtro se añade a un nodo se debe especificar un parámetro de prioridad, esto hace que el nuevo filtro se añada a una lista donde residen todos los filtros con esa misma prioridad. Cada lista de igual prioridad agrupa filtros en orden secuencial según se han añadido.

El proceso de clasificación en un nodo consiste en recorrer las listas de filtros, empezando por la lista más prioritaria. Concretamente, para procesar cada paquete se toma la lista de filtros mas prioritaria, se aplica cada filtro de esta lista secuencialmente, si el paquete cumple todas las condiciones establecidas en un filtro termina el procesado inmediatamente y se redirige el paquete un nodo indicado en el propio filtro. Si se termina una lista y no se cumple ningún filtro se repite el proceso con la siguiente lista de filtros, por orden de prioridad.

La principal restricción de los filtros es que sólo se pueden añadir a las disciplinas *classful*, es decir, las que tienen disponibles clases hijas y además, el nodo destino del filtro debe ser un nodo de tipo clase.

Salvando la restricción anterior, para los nodos restantes del árbol los filtros se pueden adjuntar a cualquiera de los restantes, pero todo paquete que llega a una interfaz el núcleo de Linux comienza su procesado aplicando únicamente las listas de filtros ubicados en la raíz[2]. El sistema operativo sólo se comunica con el nodo raíz, tanto para encolar un paquete como para obtener un paquete para su envío.

El proceso de clasificación se complica si se añaden filtros en diferentes nodos de la jerarquía, de este modo, el sistema de clasificación opera aplicando al paquete entrante los filtros del nodo raíz, estos filtros se resuelven redirigiendo a un nodo hijo, si existen filtros en el nodo hijo, se vuelven a aplicar para redirigir el paquete a otro nodo hijo de este último y así sucesivamente.

Este sistema tiene algunas restricciones bastante lógicas para evitar mal funcionamiento, un filtro asociado a un nodo sólo puede redirigir un paquete a uno de sus nodos hijos, es decir, un paquete nunca puede ser clasificado ascendentemente en el árbol.

Esta complejidad en el procedimiento de clasificación persigue el objetivo de aumentar la eficiencia del

2 No es del todo cierto, si el nodo raíz es *classless*, se considera nodo raíz de los filtros aquel nodo *classful* más cercano a la raíz del árbol.

sistema de filtrado. En contrapartida, puede llegar a ser muy complejo de administrar al estar los filtros repartidos por toda la jerarquía. Considere que la eficiencia disminuye enormemente cuando se asocia a un nodo, una lista o listas con muchos filtros, ya que se recorren todos secuencialmente. En esta situación por cada paquete hay que evaluar muchas reglas, pero si se disminuye la cantidad de reglas en un nodo y se distribuyen adecuadamente los filtros entre los nodos hijos, la cantidad reglas aplicadas al paquete hasta alcanzar su nodo destino disminuye considerablemente.

Para salvar la complejidad del sistema de filtrado, en este laboratorio se evitará asociar filtros a nodos diferentes del nodo raíz, los ejemplos de clasificación serán lo suficientemente sencillos como para no requerir optimización.

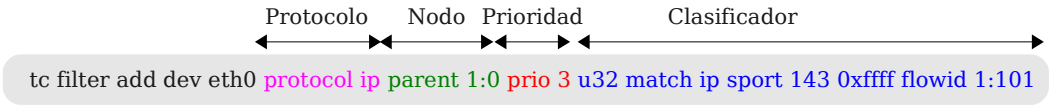

Figura 13. Ejemplo de sintaxis de filtro.

En la figura 13 se presenta con un ejemplo la sintaxis de los filtros, las partes indicadas tienen el siguiente significado:

- Protocolo: Indica el protocolo al que se aplicará el filtro, sólo se contemplará protocolo IP.
- Nodo padre: Nodo de la jerarquía al que está asociado el filtro.
- Prioridad: Indica la lista de prioridad donde se añadirá el filtro. Este número sirve para establecer la preferencia en la evaluación de los filtros, primero se evalúan las listas de filtros con un número menor, si el paquete no se clasifica entonces se proceden con las siguientes listas en orden ascendente. **Es muy importante** comenzar en 1 las prioridades, existe un error en la implementación que no considera la prioridad 0 como más prioritaria.
- Clasificador: El clasificador indica las comprobaciones a realizar al paquete y el nodo destino si el clasificador es positivo.

En la especificación de un filtro la parte más compleja es el clasificador, por existir diversidad y tener cada uno tiene su propia sintaxis. Los clasificadores más utilizados son dos: *fw* que basa la decisión en netfilter y *u32* capaz de analizar los campos paquetes a nivel de bits.

Este laboratorio se centrará en el clasificador *u32* ya que resuelve un amplio conjunto de situaciones de uso común. Otros clasificadores se pueden consultar en la documentación LARTC aunque muchos no están ampliamente documentados.

2.4.1 Clasificador u32

Es uno de los clasificadores más simples de comprender y utilizar pero su modo de operación es de bajo nivel. Con él se resuelven cantidad de posibilidades ya que selecciona partes del paquete aplicando máscaras a su contenido. Presenta cierta dificultad a la hora de interpretar las reglas que lo componen puesto que se requiere el conocimiento exacto de los campos existente del paquete que se está analizando.

Este clasificador está formado por una lista de selectores, y cada selector es un patrón que debe

cumplirse en algún campo del paquete. Si alguno de los selectores evalúa negativo entonces todo el filtro evalúa negativo. Cuando todos los selectores se cumplen en su totalidad el filtro se aplica. La sintaxis general es:

```
u32 LISTA_SELECTORES flowid M:N
```

El parámetro final *flowid* es el nodo destino del árbol si el filtro evalúa positivo y la lista de selectores puede contener tantos selectores como se deseen .Existen tres tipos de selectores y cada uno de ellos sirve para buscar un patrón en el paquete de diferente número de bits. La sintaxis general es la siguiente:

```
match u32 PATRON MASCARA at [DESPLAZAMIENTO | nexthdr+DESPLAZAMIENTO]
match u16 PATRON MASCARA at [DESPLAZAMIENTO | nexthdr+DESPLAZAMIENTO]
match u8  PATRON MASCARA at [DESPLAZAMIENTO | nexthdr+DESPLAZAMIENTO]
```

El significado de los parámetros en las tres posibilidades es el siguiente:

- u32, u16 y u8: Indican el número de bits a comprobar siendo 32, 16 y 8 respectivamente.
- Patrón: es un dato escrito en decimal o hexadecimal que debe coincidir dentro del paquete.
- Máscara: son los bits del patrón de deben coincidir.
- Desplazamiento: posición en bytes dentro del paquete donde se está haciendo la comprobación.

Seguidamente se muestran algunos ejemplos de filtros para ilustrar la sintaxis. Para interpretarlos correctamente se han incluido las figuras 15 y 16 que representan los paquetes IP con sus campos y la cabecera para el protocolo TCP también con sus correspondientes campos.

El primer ejemplo clasifica los paquetes cuyo valor TTL sea 64 al nodo 1:4:

```
tc filter add dev eth0 parent 1:0 prio 10 u32 match u8 64 0xff at 8 flowid 1:4
```

Fíjese en la figura 15 como el tamaño del campo TTL es de 8 bits y el desplazamiento en bytes del campo TTL dentro de la cabecera IP es 8.

Ahora con el siguiente ejemplo se pretende analizar los campos un protocolo IP como es TCP. El sistema de filtrado facilita la sintaxis mediante *nexthdr* para simplificar el desplazamiento, este parámetro coincide en valor con el tamaño de la cabecera IP. Así, sólo hay que indicar el desplazamiento dentro de la cabecera TCP. El ejemplo mostrado contiene ahora dos condiciones que deben cumplirse: el protocolo debe ser TCP (número 6) y el bit de ACK debe estar activo:

```
tc filter add dev eth1 parent 1:0 prio 10 u32 \
    match u8 6 0xff at 9 \
    match u8 0x10 0x10 at nexthdr+13 \
    flowid 1:3
```

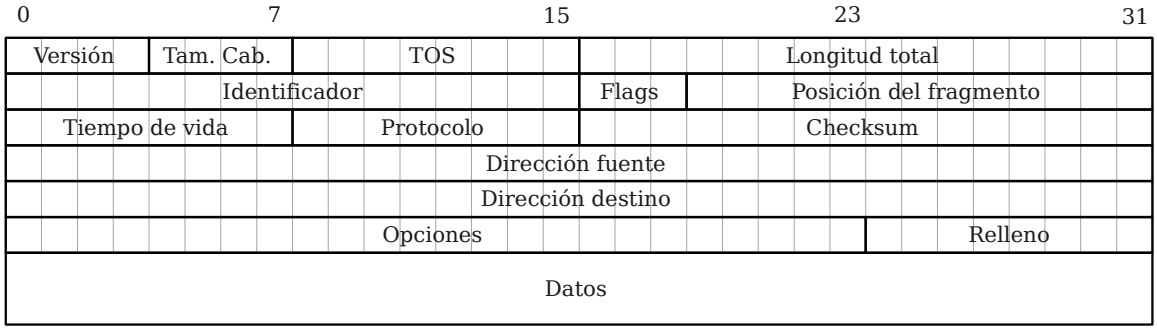

Figura 14. Campos de un paquete IP.

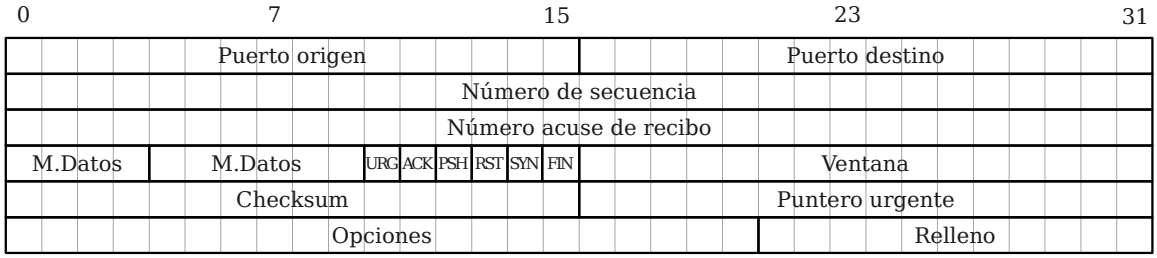

Figura 15. Campos en la cabecera de TCP.

Para facilitar el trabajo con los patrones se han existen algunos selectores específicos con nombre, con ellos no se tiene que especificar el desplazamiento de cada uno de los campos. En la tabla 14 se muestran algunos y sus equivalencias.

Selector	Equivalencia
`match ip protocol tcp`	`match u8 6 0xff at 9`
`match ip protocol udp`	`match u8 17 0xff at 9`
`match ip src 192.168.0.100/32`	`match u32 0xc0a80064 0xffffffff at 12`
`match ip dst 192.168.20.1/24`	`match u32 0xc0a81464 0xffffff00 at 16`
`match dport 21 0xffff`	`match u16 21 0xffff nexthdr + 2`
`match sport 21 0xffff`	`match u16 21 0xffff nexthdr + 0`
`match ip tos 0x10 0xff`	`match u8 0x1 0xff at 1`

Tabla 14. Selectores con nombre de mayor utilidad y sus equivalencias.

En el último ejemplo se muestra el uso de estos selectores específicos:

```
tc filter add dev eth0 parent 1:0 prio 10 u32 \
    match tcp dport 53 0xffff \
    match ip protocol 6 0xff \
    flowid 1:2
```

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 estadísticos 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 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 16.

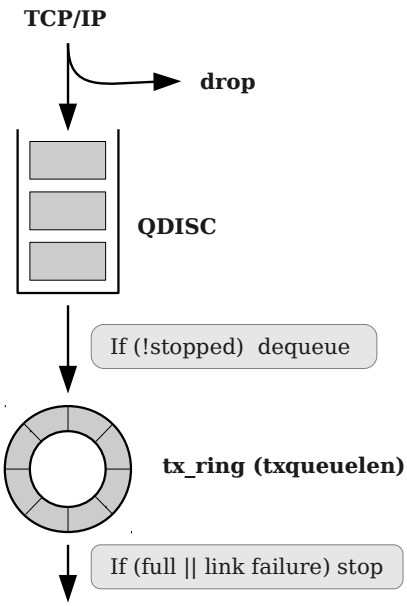

Figura 16. Colas del núcleo de Linux.

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 *ifconfig* (parámetro *txqueuelen*). Para cambiarlo sólo es posible mediante el comando *ifconfig* con la siguiente sintaxis: `ifconfig 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, sólo se podrá enviar un paquetes hasta este tamaño.

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 asignatura. Se realizará todo el conformado en la máquina que hace de puerta de enlace simulando que se dispone de un ancho de banda limitado. Todo el conformado de 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.

3.1. Soluciones con múltiples disciplinas

Esta primera parte consiste en probar diferentes disciplinas según se indica en las sucesivas tareas. Se recomienda ir guardando todos los comandos *tc* utilizados en ficheros de texto para su posterior reutilización.

Tarea 40.- Tras iniciar la máquina gateway ejecute el comando indicado como administrador para mostrar estadísticas sobre la disciplina establecida para la interfaz de forma predeterminada.

```
tc -s -d qdisc show dev eth0
```

T40.1.- Para facilitar la obtención de estadísticas cree un fichero /root/bin/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/sh
watch -n0 '
    tc -s -d qdisc show dev eth0;
    echo "-- Clases ------------------------------------";
    tc -s -d class show dev eth0'
```

T40.2.- Ejecute este script (/root/bin/estadisticas.sh) en un terminal y deje este terminal visible durante todas las pruebas realizadas posteriormente.

T40.3.- Para poder realizar pruebas de transmisión debe ubicar un fichero de al menos 10MBytes en tres ubicaciones configuradas en la sesión anterior de laboratorio: servidor WEB interno, servidor FTP interno y cuenta de usuario del gateway, esta última es para realizar transferencias por SSH.

Tarea 41.- Con la primera disciplina que se probará se pretende limitar la velocidad de salida de la interfaz exterior del entorno virtual a 150Kbits mediante una cubeta con fichas, correspondiente a la disciplina TBF.

T41.1.- Cree un nuevo fichero /root/bin/qos-t2.sh con permiso de ejecución donde 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:

```
tc qdisc del dev eth0 root
tc qdisc add dev eth0 root handle 1: tbf rate 150kbit latency 50ms burst 1540
```

T41.2.- Ejecute el script /root/bin/qos-t2.sh y observe los efectos en el terminal donde se muestran las estadísticas.

T41.3.- Desde la máquina anfitrión conecte al servidor WEB descargue un fichero observando la velocidad ¿corresponde con los 150kbits establecidos?

T41.4.- Sin detener la descarga WEB, descargue simultáneamente desde la máquina anfitrión un fichero desde el servidor FTP, puede utilizar *filezilla* ya que muestra la velocidad de descarga. ¿Se distribuye el caudal disponible equitativamente entre las dos descargas?

Junto con el material de laboratorio se adjunta un script llamado *descargas.sh* pensado para realizar varias descargas simultáneas desde el servidor WEB. Se debe usar en las tareas posteriores y uso

consiste en ejecutarlo con dos argumentos según la siguiente sintaxis: `./descargas.sh N URL` donde N es el número de descargas simultáneas y URL es la dirección completa del fichero a descargar.

Tarea 42.- Copie el fichero descargas.sh en la máquina anfitrión y establezca el permiso de ejecución.

T42.1.- Desde la máquina anfitrión pruebe realizar tres descargas simultáneamente mediante el comando:

```
./descargas.sh 3 http://192.168.20.X/fichero-grande.dat
```

T42.2.- Repita el comando anterior realizando simultáneamente 10 descargas HTTP, y simultáneamente intente acceder por FTP para descargar archivos, comprobará que las conexiones no funcionan correctamente.

T42.3.- Sin detener ninguna descarga intente acceder por SSH para obtener un terminal de texto de la máquina gateway. ¿Responde correctamente el flujo interactivo SSH con las pulsaciones de teclas?

En la situación anterior ocurren varios efectos, puede observar como se están descartando multitud de paquetes, pero el resultado habitualmente es que una de las transferencias se apodera de prácticamente todo el ancho de banda disponible, dejando paradas al resto.

Tarea 43.- Se realizará otra mejora añadiendo 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.

T43.1.- Copie el script anterior qos-t2.sh con el nombre qos-t4.sh para añadir en esté último en una nueva línea el comando:

```
tc qdisc add dev eth0 parent 1:0 handle 10: prio
```

T43.2.- 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 17.

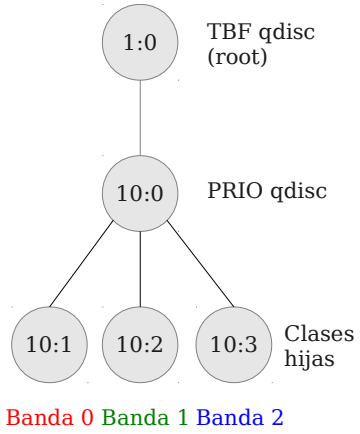

Figura 17. Disciplina PRIO.

T43.3.- Se añadirá un primer filtro para dar prioridad a los paquetes HTTP sobre el resto. Recuerde que los filtros sólo pueden añadirse a la disciplina 10:0 por ser de tipo *classful*. Añada al fichero qos-t4.sh el comando indicado, siendo cuidadoso en los retornos de carro indicados en rojo. En el fichero de script no pueden existir espacios entre la barra y el salto de línea, siguiendo este procedimiento el shell interpreta que todo el comando es una única línea.

```
tc filter add dev eth0 \↲
    protocol ip parent 10: prio 0 u32 \↲
    match ip sport 80 0xffff flowid 10:1
```

T43.4.- Realice una única descarga HTTP y en el terminal con estadística observe si los paquetes HTTP se están clasificando en la clase 10:1, debe observar si aumenta el número de bytes y paquetes enviados sólo en esa clase.

T43.5.- Pare la descarga y conecte con la máquina por SSH obteniendo un terminal, pulse teclas para observar en qué clase están clasificados los paquetes del flujo SSH.

T43.6.- Sin cerrar la conexión SSH, ejecute en la máquina anfitrión el script de descargas simultáneas con al menos 5 descargas HTTP. Cuando estén las conexiones activas intente interactuar en el terminal SSH con la máquina. ¿Es el comportamiento el esperado en estas circunstancias?

T43.7.- Simultáneamente intente acceder al FTP desde la máquina anfitrión.

Tarea 44.- Visto el comportamiento donde la cola prioritaria anula completamente el resto de flujos menos prioritarios se propone cambie el filtro anterior para intentar dar prioridad a los flujos SSH.

T44.1.- Copie el fichero `qos-t4.sh` como `qos-t5.sh` y realice dos cambios: (1) cambie el filtro HTTP para que estos flujos se clasifiquen en la clase 10:3 y (2) añada al final un nuevo filtro para que los flijos SSH se clasifiquen en la clase 10:1 (SSH usa el puerto 22)

T44.2.- Ejecute el nuevo script `qos-t5.sh` y con el script realice 10 descargas simultáneas HTTP. AL mismo tiempo compruebe que sigue respondiendo el servidor SSH de manera interactiva en un terminal.

T44.3.- Sin detener las 10 descargas conecte por SSH para descargar el fichero de gran tamaño mediante el comando *scp* o usando *filezilla*). ¿Deja el terminal SSH de responder interactivamente? ¿Siguen operando correctamente las transferencias HTTP simultáneas?

T44.4.- Intente bajo estas circunstancias usar desde una de las máquinas internas la conexión a Internet. Por ejemplo, desde la ventana de la máquina interna actualice el listado de los paquetes con *aptitude* y seguramente observará que toda la red interna ha perdido la conexión con la red exterior, el enlace de salida está saturado.

Tras las pruebas realizadas en las últimas tareas han surgido los siguientes problemas:

- Cuando varios flujos están compiten en una cola siempre alguno de ellos es dominante sobre los demás, esto se ha visto al realizar multitud de descargas simultáneas, 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.

Tarea 45.- En primer lugar se intentarán equilibrar los caudales de las transferencias simultáneas HTTP. Se añadirá una política SFQ en el árbol para tratar los flujos HTTP tal y como se muestra en la figura 18. Esta política es similar a WFQ explicada en teoría pero aplicada a flujos detectados dinámicamente.

T45.1.- De nuevo copie el fichero `qos-t5.sh` como `qos-t6.sh` y añada a este último la nueva disciplina. Añádala antes de los filtros, no al final del fichero, de la siguiente forma

```
tc qdisc add dev eth0 parent 10:3 handle 113: sfq
```

T45.2.- Pruebe en la máquina anfitrión 5 descargas

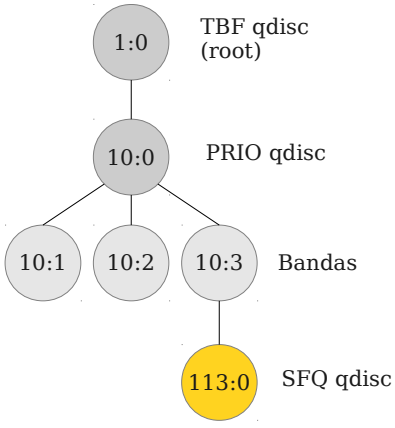

Figura 18. Disciplina SFQ.

simultáneas. Observe si todos los flujos adquieren ancho de banda de una manera más equitativa.

T45.3.- Observe en la máquina gateway la ventana de estadísticas, deberán ir apareciendo clases hijas dinámicamente por cada uno de los flujos detectados en el nodo 10:3. Cuando paren las transferencias estas clases desaparecerán.

Tarea 46.- Vuelva Copiar el script de la tarea anterior como qos-t7.sh y añada un filtro para el protocolo FTP y conseguir la clasificación indicada en la figura 19.

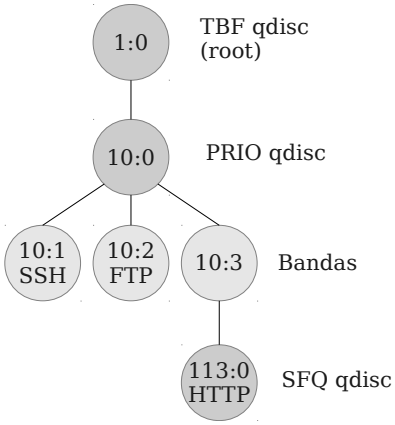

Figura 19. Ubicación de los flujos.

T46.1.- Ahora debe añadir un filtro adicional para conseguir que el trafico TCP restante se encole en la clase 10:2 mediante, el filtro es el siguiente (fíjese en la prioridad del filtro):

```
tc filter add dev eth0 \
    protocol ip parent 10: prio 99 u32 \
    match ip protocol 6 0xff \
    flowid 10:2
```

T46.2.- Ejecute 2 transferencias simultáneas HTTP, y una vez iniciadas, comience una transferencia FTP simultánea. ¿Por qué se paran 2 las transferencias HTTP?

T46.3.- Compruebe si el servicio SSH sigue operando con prioridad en estas circunstancias.

T46.4.- Inicie 10 descargas HTTP y verifique que las máquinas internas no han perdido la conexión a Internet. ¿Por qué ahora la sobrecarga HTTP no afecta a la red interna?

En la solución anterior todavía presenta algunos problemas, pueden ocurrir situaciones donde el servidor HTTP se quede sin servicio, principalmente si la clase 10:2 se satura debido al tráfico de los equipos de la red interna. Además, habrá observado que las 10 transferencias simultáneas no se sirven adecuadamente, se probarán más soluciones.

Tarea 47.- En primer lugar, para evitar que la cola 10:2 establecida como predeterminada anule el tráfico HTTP se le añadirá una disciplina TBF con un caudal máximo inferior al disponible, por ejemplo 100Kbits, así, quedan al menos 50Kbits para la siguiente cola prioritaria.

T47.1.- De nuevo copie el fichero de la tarea anterior como qos-t8.sh para trabajar en él. Añada una disciplina TBF en el árbol hija de 10:2 de la siguiente forma:

```
tc qdisc add dev eth0 parent 10:2 handle 112: \
   tbf rate 100kbit limit 10k burst 3000
```

T47.2.- Ejecute la nueva disciplina e inicie transferencias HTTP y otras FTP para comprobar si las transferencias HTTP siguen descargando, pero con menor caudal.

Tarea 48.- Termine la solución en un fichero llamado qos-t9.sh siguiendo el esquema de figura 20. Debe realizar una clasificación de los paquetes DNS con el protocolo UDP y añadir una disciplina SFQ a la cola prioritaria como la indicada.

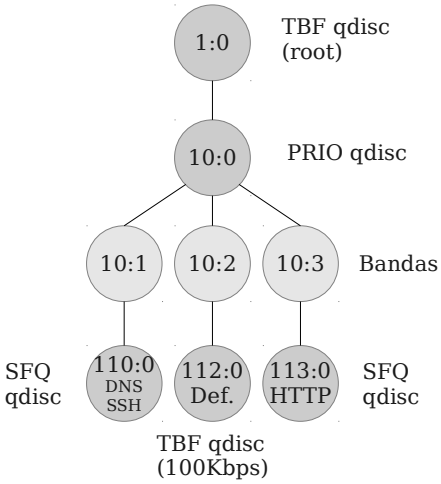

Figura 20. Ejemplo con múltiples disciplinas.

T48.1.- Una vez implementada realice 10 o 15 descargas

HTTP simultáneas ¿se equilibra el caudal de las descargas?.

T48.2.- Ahora en la máquina que ejecuta el servidor HTTP cambie la MTU a 100 mediante `ifconfig eth0 mtu 100`. Repita las 10 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 en los paquetes la relación entre los tamaños de las cabeceras y los datos enviados. 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 150kbits/seg y se deben repartir entre 10 flujos HTTP siendo la ser la MTU de 1500bytes, por tanto, los paquetes son de este tamaño. Haciendo el siguiente cálculo:

$$150\,Kbits/seg = 19200\,Kbytes/seg$$

$$\frac{19200\,Bytes/seg}{1500\,Bytes} = 12.8\,paquetes/seg \quad \Rightarrow \quad \frac{12.8\,paquetes/seg}{10\,flujos} = 1.28\,paquetes/seg$$

El resultado muestra que cada flujo tarda en recibir un paquete más de un segundo, llevando esta situación al extremo se podrían agotar los tiempos de espera de los paquetes TCP. Disminuyendo la MTU tal y como se hizo en T48.2.- se aumenta el número de paquetes por segundo que recibe cada flujo, llegando a alcanzar 10 paquetes por segundo y aumentando la fluidez. Este procedimiento es habitual cuando se necesita aumentar la interactividad de un flujo aunque se pierda eficiencia.

Finalmente, 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, y además de la complejidad, esta solución no aprovecha bien el caudal disponible, fíjese cómo en el flujo predeterminado (nodo 10:2) se ha establecido una disciplina TBF con un caudal de 100Kbps, durante el tiempo en que no existe otro tipo de tráfico sólo se utiliza 100Kbits de los 150Kbits disponibles.

Por los motivos expuestos se propone el estudio de soluciones basadas en la disciplina HTB para mejorar la eficiencia.

3.2. Implementación con HTB

HTB simplifica el procedimiento de clasificación y aprovecha en mayor medida el ancho de banda disponible. En los ejemplos propuestos se verá como es posible implementar diferentes disciplinas usando únicamente HTB con diferentes configuraciones.

Tarea 49.- Se establecerá como disciplina raíz de la interfaz una disciplina HTB. Cree un nuevo fichero qos-t10.sh para almacenar la configuración, y añada el siguiente contenido:

```
tc qdisc del dev eth0 root
tc qdisc add dev eth0 root handle 1: htb default 1
tc class add dev eth0 parent 1: classid 1:1 htb rate 150kbit burst 1500
```

T49.1.- Pruebe realizando una descarga HTTP ver si opera correctamente el límite de velocidad.

T49.2.- Realice 5 descargas simultáneas HTTP para comprobar si se equilibra la velocidad entre diferentes descargas o funcionan irregularmente como en casos anteriores.

T49.3.- Use la ventana de estadísticas para ver como HTB muestra, entre otros datos, el caudal de la clase (valor *rate*) en tiempo real.

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.

Se realizará una configuración usando únicamente en HTB y equivalente a una disciplina PRIO. Observe el parecido entre la figura 17 y lo mostrado en la figura 21. En esta última se establecen prioridades a las clases HTB consiguiendo el mismo efecto que una disciplina PRIO.

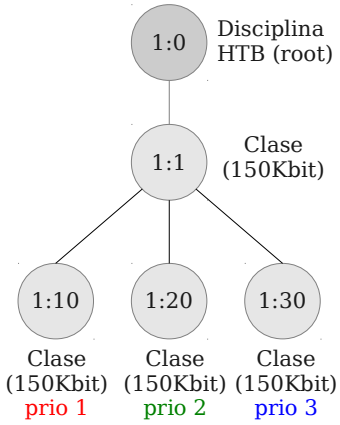

Figura 21. Jerarquía HTB equivalente a PRIO.

Tarea 50.- Para implementar la disciplina mostrada en la figura 21 use un nuevo fichero llamado qos-t11.sh, como anteriormente copie qos-t10.sh para usarlo como punto de partida.

T50.1.- En el nuevo fichero establezca para la disciplina HTB raíz como clase predeterminada la 1:30, es decir cambie: `default 30`

T50.2.- Añada tres clases hijas a 1:1 mediante

```
tc class add dev eth0 parent 1:1 classid 1:10 htb rate 150kbit prio 1
tc class add dev eth0 parent 1:1 classid 1:20 htb rate 150kbit prio 2
tc class add dev eth0 parent 1:1 classid 1:30 htb rate 150kbit prio 3
```

T50.3.- Ahora añada un filtro que clasifique todo el tráfico HTTP hacia la clase 1:10 (utilice los ejemplos ya mostrados)

T50.4.- Pruebe realizar varias descargas simultáneas HTTP y al mismo tiempo establezca una conexión SSH o FTP para verificar si los flujos HTTP tiene prioridad.

T50.5.- Realice 5 o más descargas HTTP para observar el desequilibrio en los caudales. Solucióne lo añadiendo una disciplina SFQ a la clase 1:10 y reduciendo el tamaño MTU de la máquina que contienen el servidor HTTP.

Observe como en el ejemplo mostrado las tres clases hijas tienen el mismo caudal garantizado que la clase padre, está establecido así en este ejemplo para asemejar el comportamiento a la disciplina PRIO. Aquí es donde HTB admite otra configuración alternativa que mejora considerablemente el funcionamiento de esta configuración, y lo que es más importante, es capaz de aprovechar el ancho de banda completo respecto al ejemplo mostrado en la figura 20.

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 el máximo caudal alcanzable por la

clase, pero la prioridad establece quien es el primer hijo en el reparto del ancho de banda sobrante.

Tarea 51.- Siguiendo el esquema de la figura 22 y trabajando en un nuevo fichero de nombre qos-t12.sh cambie los parámetros *rate* de las tres clases 1:1X y establezca el parámetro *ceil*.

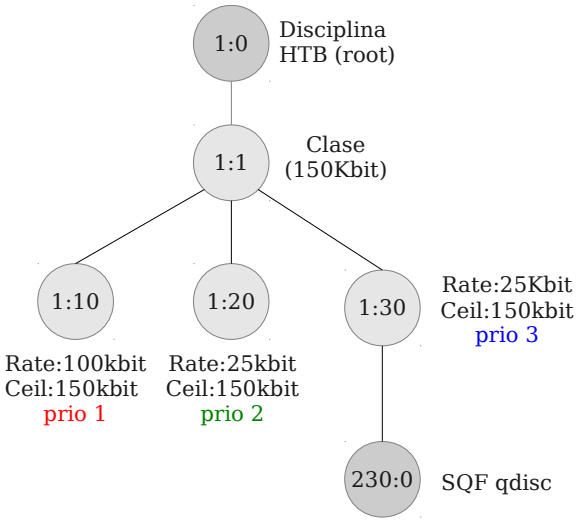

Figura 22. Jerarquía HTB con disciplinas adicionales.

T51.1.- Establezca como predeterminada la clase 1:20 y clasifique todo el tráfico HTTP en la clase 1:30.

T51.2.- Clasifique todo el tráfico SSH en 1:10.

T51.3.- Realice 10 descargas simultáneas HTTP pruebe si la conexión SSH responde de manera interactiva.

T51.4.- Anule las 10 descargas y realice una única descarga HTTP para medir la velocidad. Al mismo tiempo realice otra transferencia FTP ¿garantiza la clase 1:30 25kbit al tráfico HTTP? ¿Cual 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.

4. Estudio no guiado

Para finalizar el laboratorio se propone realizar un ejercicio no guiado usando principalmente la disciplina HTB. Concretamente, el ejemplo mostrado en la figura 23 corresponde a una posible configuración para un caudal típico de salida de una conexión DSL donde se desea dar diferentes servicios y sólo se controla la frontera de red.

Aunque pueda parecer extraño la existencia de las dos primeras clases se debe a que el equipo que realiza el confirmado del tráfico está conectado con una única interfaz de red, y en ella, conviven la red interna y el router DSL. Todo el tráfico con destino a Internet está clasificado en la rama 1:2, mientras que el tráfico de la red interna se clasifica en la rama 1:1 aprovechando al máximo el caudal disponible en ethernet.

Realice la implementación según las indicaciones siguientes:

Tarea 52.- Implemente la jerarquía HTB mostrada en la figura 23 con las siguientes consideraciones.

T52.1.- Todo el tráfico de la red interna debe clasificarse en la clase 1:1, use una regla prioritaria para clasificar todas las direcciones de la red 192.168.0.0/24 en esta clase.

T52.2.- El servicio DNS se refiere a la peticiones DNS realizadas a servidores externos, no se dispone de un servidor DNS propio.

T52.3.- El protocolo SMTP se refiere a conexiones salientes hacia servidores de correo.

T52.4.- La clase 1:99 es la clase predeterminada.

T52.5.- Considere que debe añadir un servicio VPN considerado como prioritario ¿A que clase lo añadiría?, añada un filtro que lo clasifique según su decisión.

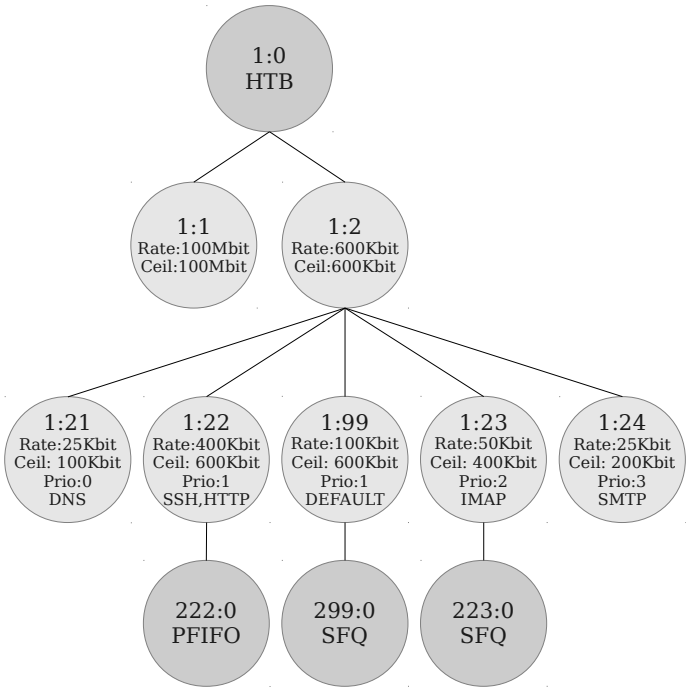

Figura 23. Ejemplo de jerarquía HTB con múltiples servicios.

Tarea 53.- Puede realizar algunas pruebas opcionales no estudiadas en este laboratorio

T53.1.- Experimente con la disciplina CHOKE colapsando algún servicio. ¿Para que sirve esta disciplina?

T53.2.- Compruebe que SFQ no es buena política para páginas WEB, si la página tiene mucho contenido (muchas imágenes) ocurre un efecto extraño, se cargan todas la imágenes en paralelo y lentamente. Puede probar una simple PFIFO con un tamaño de un solo paquete.

T53.3.- Pruebe alterar el tamaño de cola de la interfaz para ver si afecta a los resultados, establezca valores grandes y pequeños con *ifconfig* y altere el tamaño de ráfaga en la clase 1:2 y 1:1.