Aplicación de la Virtualización al estudio de técnicas de Calidad de Servicio - QoS

Juan Quiros*, Paulino Ruiz-de-Clavijo†, Alejandro Carrasco‡, Julian Viejo§ and Alejandro Millan¶

*†§¶ Grupo ID2 (Investigación y Desarrollo Digital)
ETSI Informática (Tec. Electrónica)
Universidad de Sevilla, Sevilla 41012, Spain
www.dte.us.es/id2

‡ Electronic Technology and Industrial Informatics
ETSI Informática (Tec. Electrónica)
Universidad de Sevilla, Sevilla 41012, Spain
www.dte.us.es/id2

*jquiros@dte.us.es, †paulino@dte.us.es, ‡acarrasco@us.es, §julian@dte.us.es, ¶amillan@us.es

***Resumen*—Este trabajo abarca la aplicación de técnicas de calidad de servicio en redes de conmutación de paquetes. Para ello se presenta una metodología basada en técnicas de virtualización. Con ello se aporta al alumno un enfoque muy práctico y un entorno fácilmente accesible, todo ello desde un punto de vista adecuado a la titulación de Grado en Ingeniería Informática, Tecnologías de la Información.**

I. Introducción

El trabajo presentado se incluye dentro de la docencia impartida en la asignatura de Tecnologías Avanzadas de la Información, asignatura de tercer año del Grado en Ingeniería Informática - Tecnologías Informáticas de la Universidad de Sevilla.

A lo largo del documento se presenta la experiencia docente de la aplicación de un nuevo programa de laboratorio paralelo al programa teórico. Este programa de laboratorio, además de mantener la coherencia con el anterior, persigue la aplicación de tecnologías en auge, como son la virtualización y el *cloud computing* junto con la aplicación de técnicas de calidad de servicio.

El programa de laboratorio se plantea de forma incremental, es decir, aunque cada laboratorio tiene unos objetivos concretos, en su conjunto todos los laboratorios construyen un sistema completo con una determinada arquitectura de red. Finalmente, usando dicha arquitectura y una serie de servicios, se trabaja sobre técnicas de calidad de servicio para redes de conmutación de paquetes.

El trabajo está organizado como sigue: en el siguiente apartado se presenta la metodología docente que se sigue en la asignatura. Posteriormente se analiza el desarrollo práctico de la misma, y finalizaremos mostrando los resultados y conclusiones más relevantes obtenidos en los dos años de impartición de la asignatura.

II. Objetivos y metodología

El objetivo principal del programa de laboratorio es dotar a los egresados de los conocimientos necesarios para la puesta en marcha de los sistemas informáticos requeridos por cualquier organización. Se toman como base los conocimientos básicos de *networking*, arquitectura de computadores, ingeniería del software y sistemas operativos, adquiridos por el alumnado en cursos anteriores. En este contexto, el principal objetivo establecido en la asignatura es el adquirir los conocimientos necesarios relativos a:

- El despliegue de infraestructuras de red a nivel lógico y de aplicación.
- Seguridad en redes de computadores.
- Servicios avanzados de red, así como su ordenación mediante técnicas de prioridad y/o calidad de servicio.

Adicionalmente a estos objetivos, transversalmente se contempla la introducción a la administración de sistemas Unix y el uso de tecnologías de virtualización. Al ser esta última una tecnología con un amplio ámbito de aplicación, en la asignatura se usa en el campo de arquitectura de redes y despliegue de servicios. El uso de esta tecnología pretende aportar dos visiones al alumno: realidad de mercado respecto a arquitecturas *cloud* y facilidad de uso de entornos flexibles con pocos recursos.

Para lograr los objetivos anteriormente descritos, el contenido teórico de la asignatura se estructura en cuatro bloques. El primero de ellos se centra en los fundamentos de la seguridad en redes: conceptos generales, principales problemas y amenazas y seguridad perimetral [1], [2]. El segundo atiende a la interconexión segura de equipos, profundizando en las *Virtual Protect Networks* (VPN) [3]. En el siguiente bloque se estudia y analiza la gestión del tráfico y de la calidad de servicios QoS en redes [4], [5], [6], centrándose en el control del ancho de banda y el marcado de paquetes. Por último, se enumeran algunos servicios avanzados de red, con el objetivo de que el desarrollo de este punto lo lleve a cabo el alumno de forma tutorizada y con el apoyo de las sesiones de laboratorio, profundizando en el servicio que prefiera.

II-A. Estructura de laboratorios

Los laboratorios de la asignatura complementan los conocimientos adquiridos en la parte teórica, presentando una estructura muy similar. Se dividen en cinco bloques, pensados para desarrollarse en el orden indicado. Cada sesión se inicia con una introducción de las tecnologías a emplear, completando los conocimientos teóricos impartidos previamente; seguidos

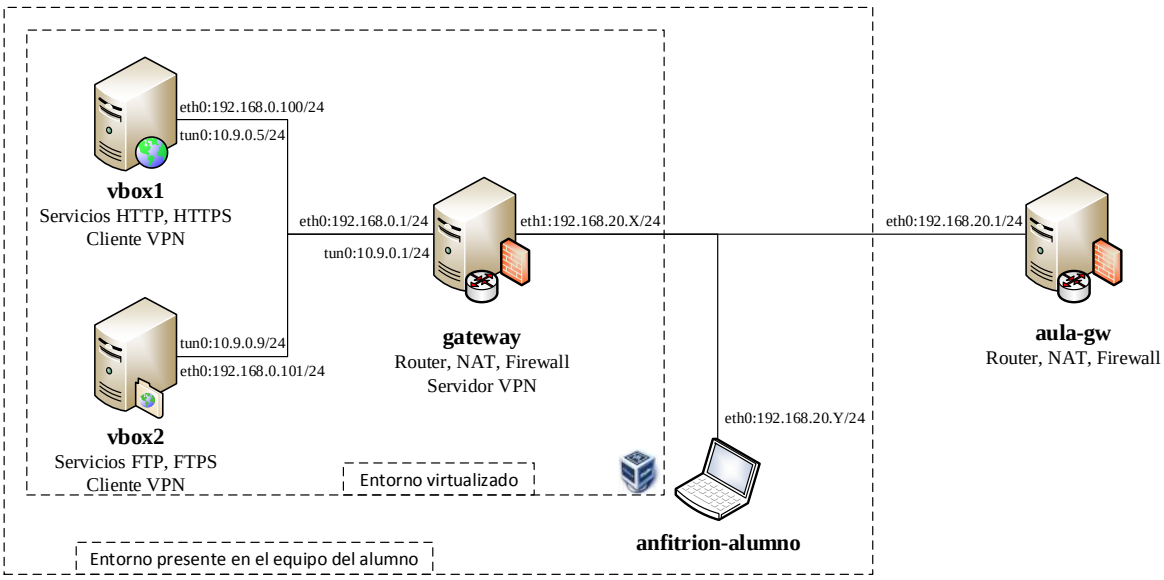

Figura 1. Esquema de configuración de la red virtual.

de una parte guiada, con el fin de que el alumno se familiarice con las herramientas; y concluye con una parte no guiada, que permite afianzar y profundizar en los conocimientos adquiridos, así como en su aplicación práctica.

Todos los bloques se desarrollan en el contexto de la arquitectura de laboratorio presentada en la Fig. 1, en la que se distinguen dos redes: la *192.168.0.0/24*, privada para cada alumno, y la *192.168.20.0/24*, común para todo el aula. La red *192.168.20.0/24* es una red virtual montada sobre la propia red física empleada en las aulas de laboratorio. Tiene acceso a internet a través del equipo *aula-gw*, configurado como puerta de enlace. Cada alumno tiene reservada dos direcciones consecutivas de esta red, una para el equipo *gateway* y otra para su equipo anfitrión, *anfitrión-alumno*. De este modo, es posible crear la infraestructura necesaria para la realización de los laboratorios empleando los recursos existentes y minimizando su impacto en estos últimos. La red privada, junto con los equipos que la componen (*vbox1*, *vbox2* y *gateway*), está virtualizada empleando el software VIRTUALBOX [7], de ORACLE. De ese modo, las sesiones de laboratorio no precisan de las infraestructuras del aula, por lo que el alumno puede trabajar en otros entornos, por ejemplo desde su casa. Para ello, únicamente tendría que modificar la configuración de red de su equipo, *anfitrion-alumno* y de la interfaz *eth1* de la máquina virtualizada *gateway*.

El propósito del primer bloque es la preparación de una arquitectura básica de red, Fig. 1, que se enriquecerá con nuevos servicios y características a lo largo de las sucesivas sesiones. Haciendo uso de la tecnología de virtualización se crean las tres máquinas virtuales con el Sistema Operativo (SO) UBUNTU SERVER 12.04 [8]. Estas máquinas son configuradas según el esquema de red anteriormente presentado.

El siguiente bloque complementa los conceptos de seguridad vistos en la parte teórica, y tiene como objetivo la implementación del cortafuegos en la máquina virtual *gateway*, Fig. 1, empleando la herramienta IPTABLES, componente del *framework* NETFILTER del núcleo de LINUX [9], [10]. Para ello, se introduce su funcionamiento y uso, haciendo énfasis en el filtrado de paquetes, traducción de direcciones (*Network Address Translation*, NAT) y creación de *log*.

El tercer laboratorio se corresponde con el segundo bloque teórico. Su propósito es el de crear y configurar una VPN [3] en el entorno virtual de desarrollo, creando las interfaces *tun0* en los equipos virtualizados (Fig. 1). Para ello se hace uso de la tecnología OPENVPN de OPENVPN TECHNOLOGIES, INC [11].

En la cuarta sesión se añaden los servicios HTTP y HTTPS a la máquina *vbox1* y FTP y FTPS a la máquina *vbox2*, Fig. 1. Se concede plena libertad a los alumnos para la selección, instalación y configuración de los recursos software que implementen estos servicios. Además se debe configurar el cortafuegos de la máquina *gateway* para permitir el acceso a estos recursos desde el exterior. El objetivo de esta práctica es dotar a la arquitectura de red de los servicios necesarios para desarrollar la última sesión de laboratorio.

Por último, en el quinto laboratorio se completa la arquitectura de red con mecanismos de control de tráfico y calidad de servicio, desarrollando de forma práctica el tercer bloque teórico. Este último laboratorio es el más extenso de todos, abarcando aproximadamente el 40 % de todo el programa.

Tomando como base todos los conocimientos adquiridos en el transcurso del curso, la asignatura finaliza con un trabajo a desarrollar por parte del alumno, consistente en la adición de un servicio avanzado a la arquitectura de red completada a lo largo de las sesiones de laboratorio, Fig. 1. Algunos de ellos son una auditoría de seguridad sobre la arquitectura de laboratorio, la configuración de un proxy transparente e inverso con servicios de balanceo de carga, monitorización de servicios de red empleando herramientas como NAGIOS [12] y MUNIN [13] e instalación, configuración y comparación

de servicios de almacenamiento en la nube: BTSYNC [14] y OWNCLOUD [15].

III. CALIDAD DE SERVICIO

LINUX ofrece gran multitud de mecanismos de control de tráfico. Sin embargo, su uso y configuración presentan cierta complejidad, debido al elevado número de alternativas, su escasa documentación y su enfoque centrado en el desarrollo e incorporación al núcleo de nuevos módulos, frente a la usabilidad. Dada la gran variedad de módulos, han sido seleccionados los más utilizados en entornos reales.

III-A. Fundamentos teóricos

El control de tráfico en LINUX [6] 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, que pueden llevar asociada una o varias colas, de modo que 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. Así, 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 cola se asocian jerárquicamente formando una estructura en forma de árbol, que toma cono nodo raíz la interfaz de red. Posteriormente, a ciertos nodos se adjuntan reglas, denominadas filtros, para decidir cual es el nodo destino de cada paquete. Algunas disciplinas van más allá de ser un simple nodo de la jerarquía, incluyendo subnodos o clases hijas, donde se pueden encolar los paquetes. En la Fig. 2 se muestra un esquema con una posible configuración con diferentes disciplinas y clases. Para entender este mecanismo, es necesario profundizar en los tres tipos de elementos que forman parte del árbol: disciplinas, clases y filtros.

Las clases siempre forman parte de una disciplina, no pueden existir fuera de ellas, siendo su principal uso el de la clasificación de los paquetes. De ese modo, si 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 única disciplina se consigue esta configuración.

Respecto a las disciplinas, éstas poseen 4 funcionalidades: aceptar los datos, reordenar su envío, retrasarlo o eliminarlo de la cola. Además, no todas las disciplinas implementadas disponen de clases hijas, distinguiéndose dos tipos: disciplinas sin clasificación (*classless*) y disciplinas con clasificación (*classful*).

Atendiendo a los filtros, se definen como reglas de clasificación. En este punto, es preciso conocer como el núcleo de Linux interactúa con el árbol jerárquico de disciplinas creado por el administrador del sistema. El punto de entrada de los paquetes es siempre la raíz de la jerarquía. Así, cuando los paquetes llegan a una interfaz de red, se les aplican los filtros asociados a la raíz de la jerarquía. De ese modo, a través de la evaluación de los filtros los paquetes se encolan en una clase concreta del árbol. También es posible asociar filtros a otros nodos internos, obteniendo caminos con múltiples puntos de decisión. Esta segunda solución presenta una mayor complejidad, y únicamente está indicada en situaciones donde existen multitud de filtros por cuestiones de eficiencia.

Tal y como se comentó anteriormente, las disciplinas de colas se estructuran jerárquicamente formando un árbol, donde cada nodo se corresponde con una clase o una disciplina. A la hora de asociar un filtro, es necesario poder identificar unívocamente cada uno de los nodos, con el fin de poder ubicar cada paquete en el nodo correspondiente. Este identificador es una pareja de números llamados número mayor y número menor, separados por ":", y serán asignados explicita o implícitamente durante la configuración. Los números mayores siempre identifican las disciplinas, así, cada disciplina tendrá un número mayor distinto. En cambio, el número menor identifica a cada clase hija (de una disciplina), siendo el número mayor el mismo para todas las clases hijas de la misma disciplina.

Una vez introducidos los principales conceptos, pasaremos a enumerar las disciplinas más relevantes. De la diversidad de disciplinas implementadas en el núcleo de Linux se han escogido las siguientes para su estudio en el laboratorio.

- **PFIFO y BFIFO:** Disciplinas simples basadas en una cola FIFO (*First In First Out*) 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.
- **SQF (*Stochastic Fairness Queueing*):** Disciplina llamada estocástica equitativa basada en la autodetección de flujos, reordena los paquetes para tratar equitativamente a cada flujo.
- **TBF (*Token Bucket Filter*):** Disciplina de cubeta con fichas para la regulación de la velocidad.

Las disciplinas con clasificación seleccionadas para el laboratorio son las siguientes:

- **PRIO:** Disciplina de cola con bandas de prioridad configurables, no implementa regulación de velocidad.
- **HTB (*Hierarchy Token Bucket*):** Disciplina que comparte el ancho de banda entre clases hijas, garantizando un ancho de banda en cada clase, además de priorizar y regular la velocidad del mismo modo que la disciplina sin clasificación TBF.

Además de las enumeradas anteriormente, se propone al alumno el estudio y prueba de otras existentes de manera opcional como son: CHOKE (*CHOose and KEep for responsive flows*), CBQ (*Class Based Queueing*), DRR (*Deficit Round Robin Scheduler*) y RED (*Random Early Detection*)

III-B. Herramientas de configuración

Aunque existe una gran variedad de implementaciones de disciplinas, todas ellas presentan un modo de configuración homogéneo, a través de una única herramienta, TC [6]. Por lo tanto, la sintaxis y un conjunto de opciones son comunes a cualquier implementación y, para cada disciplina, únicamente

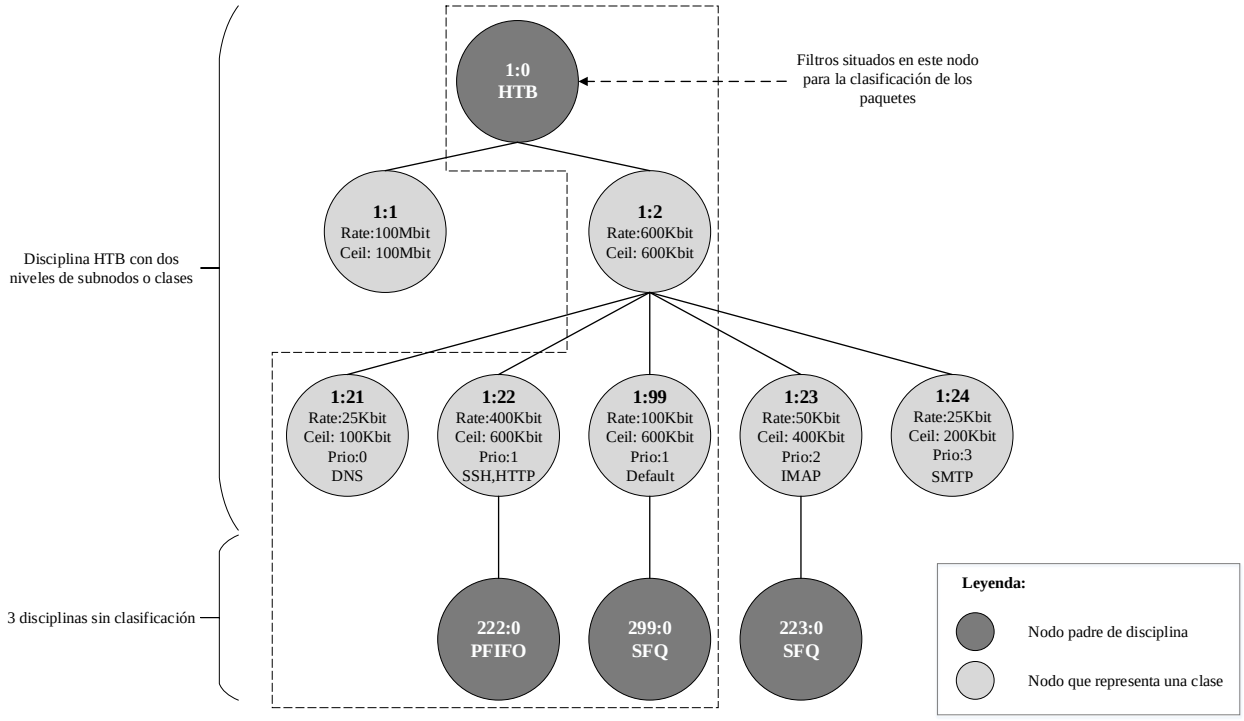

Figura 2. Ejemplo de jerarquía HTB típico de una conexión ADSL.

es necesario introducir aquellos parámetros de configuración específicos.

El comando TC posee tres modos, cada uno de ellos destinado a manipular un elemento diferente del sistema de control de tráfico: disciplinas, clases y filtros. Dado que su explicación en detalle trasciende los objetivos de este artículo, su sintaxis y uso se mostrarán a través de ejemplos.

En el modo de configuración de disciplinas, además de especificar la disciplina y su configuración específica, se debe indicar el identificador del nodo y su nodo padre en la jerarquía. Algunos ejemplos se muestran en el Código 1, líneas 7 (HTB en la raíz de la jerarquía), 21-22 (SFQ) y 36-37 (PFIFO), que hace referencia a la Fig. 2.

El modo de configuración de clases únicamente puede ser usado cuando se está configurando una disciplina con clasificación. Su sintaxis es muy similar a la empleada para las disciplinas, aunque con algunas particularidades. Para la disciplina HTB mostrada en la Fig. 2, se pueden ver algunos ejemplos de configuración en el Código 1, líneas 11-12, 19-20, 27-28 y 34-35.

A diferencia de la sintaxis para clases y disciplinas, los filtros mantienen una sintaxis similar, independientemente de la disciplina que se esté configurando. Los filtros se adjuntan a determinados nodos de la jerarquía de disciplinas con clasificación, constituyendo una lista, que tiene asociada una prioridad representada por un número natural, tomando 1 como la más prioritaria. Así, cuando un filtro se añade a un nodo se debe especificar un parámetro de prioridad, añadiéndose a la lista donde residen todos los filtros de igual precedencia. De ese modo, dados todos los filtros de un nodo, se evalúan y se aplican, si corresponde, en primer lugar aquellos localizados en listas de mayor prioridad y, dentro la misma lista, siguiendo un orden secuencial de inclusión.

El clasificador se compone de dos partes diferenciadas: un conjunto de comprobaciones o patrón de búsqueda, y una acción a realizar sobre el paquete que encaje en el patrón, generalmente el encaminarlo a un nodo de la jerarquía. El filtro nos permite clasificar mensajes según el contenido del paquete de nivel 3 del modelo OSI, por ejemplo un determinado valor en un campo de la cabecera IP. En la especificación de un filtro, el clasificador constituye la parte más compleja, debido a la variedad de opciones disponibles, cada una con una sintaxis propia. Los clasificadores más utilizados son dos: FW, que basa la decisión en netfilter y *u32*, capaz de analizar los campos de los paquetes a nivel de bits, aplicando máscaras y desplazamientos. Se pueden ver algunos ejemplos de filtros empleando el clasificador U32 en el Código 1, líneas 39-48.

III-C. Contenido de la sesión de laboratorio

El objetivo de la sesión de laboratorio es la puesta en práctica, de forma directa, de los conocimientos adquiridos, así como el realizar pruebas en entornos reales, y no únicamente en el entorno virtualizado. Para ello y, atendiendo a la arquitectura de red montada a lo largo de las anteriores sesiones, el alumno debe aplicar a la interfaz *eth1* de la máquina *gateway*, Fig. 1, distintas políticas de calidad de servicio, culminando con una configuración más compleja, que se muestra en la Fig. 2. Esta última política es de aplicación directa en conexiones ADSL en entornos domésticos, y tiene como objetivo regular

el tráfico saliente a internet, asegurando el servicio DNS y dando prioridad al tráfico HTTP y SSH en primer lugar, seguido de otros protocolos no especificados y, por último, el tráfico IMAP y SMTP, en ese orden.

A continuación se muestra el código necesario para configurar la interfaz *eth1* de la máquina *gateway*, según el fragmento señalado del árbol mostrado en la Fig. 2:

Código 1. Configuración, para la interfza *eth1* de la máquina *gateway*, del fragmento seleccionado del árbol de la Fig. 2.

```
# Borrar una configuración anterior.
tc qdisc del dev eth1 root

#### Estructura de árbol (clases y disciplinas) ####
#Configuración de disciplina HTB en nodo raíz.
#Lo no clasificado toma la clase id 1:99
tc qdisc add dev eth1 root handle 1: htb default 99

#Clase id 1:2 configurada para tráfico hacia
#internet. Ancho de banda máximo de 600Kbit.
tc class add dev eth1 parent 1:0 classid 1:2 htb \
    rate 600 ceil 6000 burst 3000

#Clases id 1:99 y 299:0 configuradas para tráfico
#no clasificado. Ancho de banda asegurado de 100Kbit
#y máximo de 600Kbit.
#Al tráfico se le aplicará,
#adicionalmente, una disciplina SFQ.
tc class add dev eth1 parent 1:2 classid 1:99 htb \
    rate 100kbit ceil 600 quantum 5000 prio 1
tc qdisc add dev eth1 parent 1:99 handle 299: sfq \
    perturb 5

#Clase id 1:21 configurada para tráfico DNS.
#Ancho de banda asegurado de 25Kbit y máximo
#de 100Kbit.
tc class add dev eth1 parent 1:2 classid 1:21 htb \
    rate 25kbit ceil 100kbit prio 0

#Clases id 1:22 y 222:0 configuradas para tráfico
#SSH y HTTP. Ancho de banda asegurado de 400Kbit y
#máximo de 600Kbit. Al tráfico se le
#aplicará, adicionalmente, una disciplina PFIFO.
tc class add dev eth1 parent 1:2 classid 1:22 htb \
    rate 400kbit ceil 600 prio 1
tc qdisc add dev eth1 parent 1:22 handle 222: pfifo \
    limit 100

#### Filtros de clasificación ####
#Tráfico HTTP y SSH hacia nodo id 1:22
tc filter add dev eth1 protocol ip parent 1:0 prio 2\
    u32 match ip sport 80 0xffff flowid 1:22
tc filter add dev eth1 protocol ip parent 1:0 prio 2\
    u32 match ip sport 22 0xffff flowid 1:22

#Tráfico DNS hacia nodo id 1:21
tc filter add dev eth1 protocol ip parent 1:0 prio 2\
    u32 match ip dport 53 0xffff flowid 1:21
```

Por último, una vez aplicada una política de QoS, el alumno puede comprobar su funcionamiento a través de un *script* que, haciendo uso de la herramienta TC, muestra las estadísticas de los distintos nodos del árbol. En la Fig. 3 se muestra la salida, en un instante determinado, de las estadísticas para el ejemplo de jerarquía HTB de la Fig. 2. Adicionalmente, algunos alumnos han comprobado su funcionamiento en un entorno doméstico real, con tráfico generado en su propia red privada.

Figura 3. Terminal con las estadísticas del enlace empleando la herramienta TC. Las disciplinas están rodeados por un cuadrado y las clases por una elipse.

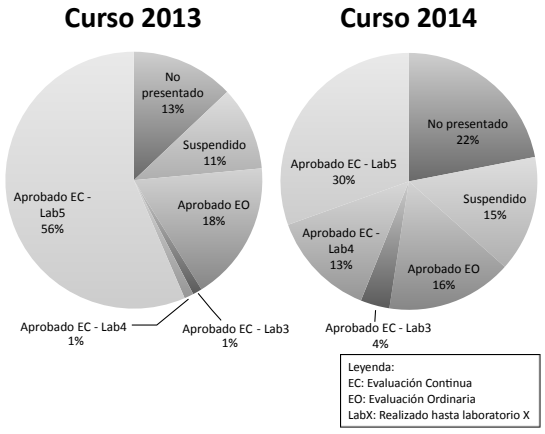

Figura 4. Resultados académicos del curso 2012/13 y 2013/14.

IV. Resultados

Tras dos años de docencia, se recogen los resultados académicos obtenidos por los alumnos, mostrados en la Fig. 4, así como el número de laboratorios realizados. Existen dos vías para superar la asignatura: evaluación ordinaria, en la que los alumnos deben presentarse a un examen teórico y práctico, y evaluación continua, en la que se sigue la metodología presentada en el artículo. En este sentido, para superar los laboratorios, es necesario realizar los 3 primeros bloques. En el curso 2013, el $58,5\,\%$ aprobó siguiendo la metodología aquí descrita, suponiendo el $76,9\,\%$ de los aprobados totales ($87,8\,\%$ de los presentados), frente al $47,6\,\%$ del curso 2014, que supusieron el $75\,\%$ de los aprobados totales de ese año ($81,3\,\%$ de los presentados). Se puede apreciar que se mantiene el porcentaje de aprobados empleando esta metodología respecto a los aprobados totales. El descenso de los aprobados totales se debe a la transición a los nuevos planes de estudio: el año pasado gran cantidad de alumnos provenían del plan antiguo, con mayor conocimientos de redes de computadores, debido a una estructuración distinta de las asignaturas, unido al hecho de que muchos de ellos tenían aprobadas asignaturas de cursos superiores, con gran parte de los conceptos ya interiorizados.

Con respecto al grado de completitud de las sesiones de laboratorio, destaca que en el curso $2012/13$ un $56\,\%$ de los alumnos totales realizaron, completa o parcialmente el último bloque de calidad de servicio. Este resultado ha descendido en el último curso a un $30\,\%$ por los motivos mencionados anteriormente.

Por último, respecto a las principales dificultades encontradas por los alumnos, destaca especialmente la barrera de entrada que supone el uso de consola en la administración de sistemas UNIX, especialmente cuando el único medio de interacción empleado mayoritariamente por el alumnado es la interfaz gráfica de Microsoft Windows, siendo marginal el uso de la consola de sistema.

V. Conclusiones

Con el programa de laboratorio presentado en este trabajo se ha conseguido aportar al alumno un punto de vista mucho más práctico en el ámbito de arquitectura de redes, virtualización y calidad de servicio dentro del marco de la titulación.

Además se ha desarrollado el programa de laboratorio empleando elementos que son muy accesibles, como es VirtualBox y Linux, facilitándole el trabajo al alumno en entornos ajenos al laboratorio.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Ministerio de Ciencia e Innovación del Gobierno de España a través del proyecto TEC2011-27936 (HIPERSYS), por el Fondo Europeo de Desarrollo Regional (FEDER), y por el Ministerio de Educación del Gobierno de España a través de la beca AP2009-3625 del programa de becas FPU.

Referencias

[1] A. Villalon Huerta, "Seguridad en unix y redes," http://www.rediris.es/cert/doc/unixsec/, 2002.

[2] S. Northcutt, L. Zeltser, S. Winters, K. Kent, and R. W. Ritchey, *Inside Network Perimeter Security (2Nd Edition) (Inside)*. Indianapolis, IN, USA: Sams, 2005.

[3] M. Feilner, *OpenVPN: Building and Integrating Virtual Private Networks*. Packt Publishing, 2006. [Online]. Available: http://portal.acm.org/citation.cfm?id=1202604

[4] J. W. Evans and C. Filsfils, *Deploying IP and MPLS QoS for Multiservice Networks: Theory & Practice*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[5] J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 6th ed. USA: Addison-Wesley Publishing Company, 2009.

[6] B. Hubert, "Linux advanced routing & traffic," http://www.lartc.org/, 2014.

[7] Oracle, "Virtualbox web page," https://www.virtualbox.org/, 2014.

[8] Canonical, "Ubuntu server web page," http://www.ubuntu.com/server, 2014.

[9] R. Russell, "Linux networking-concepts," http://www.netfilter.org/documentation/, 2001.

[10] M. Rash, *Linux firewalls : attack detection and response with iptables, psad, and fwsnort*. San Francisco: No Starch Press, 2007, index. [Online]. Available: http://opac.inria.fr/record=b1124526

[11] M. Feilner, *Beginning OpenVPN 2.0.9: Build and Integrate Virtual Private Networks Using OpenVPN*, ser. From technologies to solutions. Packt Publishing, Limited, 2009. [Online]. Available: http://books.google.es/books?id=SLlxmAEACAAJ

[12] W. Barth, *Nagios. System and Network Monitoring*. No Starch Press, 2006.

[13] B. t. Brinke, *Instant Munin Plugin Starter*. Packt Publishing, 2013.

[14] B. Inc, "Bittorrent sync user guide," http://btsync.s3-website-us-east-1.amazonaws.com/BitTorrentSyncUserGuide.pdf, 2014.

[15] ownCloud, "owncloud administrators manual," http://doc.owncloud.org/server/6.0/admin_manual/, 2014.