Sirviendo web desde la escuela

Antonio Saorín

saorin@teleline.es

José J. Grimaldos

josejuan.grimaldos@hispalinux.es

Ginés Ángel Esteban

ginees@larural.es

José Antonio Morgado

jamorgadoes@yahoo.es

María del Mar Lera

José Luis López-Barajas

Sara Redondo

Sirviendo web desde la escuela

por Antonio Saorín, José J. Grimaldos, Ginés Ángel Esteban, José Antonio Morgado, María del Mar Lera, José Luis López-Barajas, y Sara Redondo

Historial de revisiones

Revisión 0.1 12-03-2003 Revisado por: jjg

Redacción inicial en DocBook SGML usando el editor Emacs

Tabla de contenidos

Capítulo 1. Introducción

Este documento surge fruto de un grupo de trabajo en el I.E.S. Cura Valera (http://www.iescuravalera.org) de Huércal-Overa integrado por profesores del mismo, con la intención de elaborar una pequeña guía que permita a cualquier centro educativo, conectado a la red Internet mediante ADSL, servir sus páginas web desde una máquina local.

Este grupo de trabajo quedó registrado con el código 02403GT120 en el Centro de Profesores Cuevas-Olula (http://averroes.cec.junta-andalucia.es/~cepal3) durante el curso escolar 2002/2003

Pretendemos documentar una experiencia llevada a cabo en nuestro centro y, a través de ella, conseguir que todo aquel interesado cuente con un material que le permita soslayar muchas de las dificultades con que nos encontramos nosotros inicialmente al comienzo de nuestro proyecto.

No pretende, en ningún caso, ser una documentación de referencia para ninguno de los apartados tratados en él, muchos de los cuales poseen ya varios volúmenes monográficamente dedicados. Procuraremos, sin embargo, dejar las pistas necesarias para un estudio más detallado y profundo cuando el lector así lo requiera.

Pensamos, además, que esta pequeña guía sirva para sugerir las enormes posibilidades que brinda la utilización de software libre en los entornos educativos, frente a la realidad actual, donde en la mayoría de casos se utiliza y se forma al alumnado en sistemas propietarios, fomentando así una situación nada deseable en el mundo de la educación, en particular, de la educación pública.

Aunque suponemos, al menos, cierta inquietud y cierta familiaridad con el mundo de la informática, trataremos de mostrar los procedimientos de la forma más sencilla posible con objeto de que cualquier lector interesado sea capaz de seguirlos, de modo que, pedimos perdón por aquellas obviedades o presunciones en que podamos incurrir.

Capítulo 2. Conceptos de redes

2.1. Preliminares

No hace tanto tiempo que pensar en una red de ordenadores era poco menos que un sueño. Estábamos tan bien con nuestra computadora personal o laboral, éramos capaces -creíamos- de procesar texto, manejábamos nuestras bases de datos, imprimíamos vistosos carteles ahítos de alardes tipográficos,... en fin. Hoy día, en cambio, no se concibe una computadora aislada. La más modesta dispone de un artefacto capaz de comunicarse con otras a través del hilo telefónico. Hoy en día tenemos que estar en *red*.

Consideraremos una red como un conjunto de ordenadores capaces de comunicarse entre sí, bien directamente, bien a través de otros.[1] Como en toda comunicación, para que ésta sea posible, necesitamos un *idioma* que sea comprendido por todos los integrantes, en este caso, los ordenadores de la red. En este contexto el idioma es el *protocolo de comunicación*

2.2. Protocolo TCP/IP

Muchos son los protocolos usados desde siempre para compartir datos entre diferentes ordenadores, sin embargo, desde la aparición de la red Internet, todo parece normalizarse en torno al TCP/IP.[2]

Para ser exactos no se trata de un protocolo sino de un conjunto de protocolos, aunque este tipo de disquisiciones se escapan al objetivo de esta guía. Para un estudio más detallado de las redes debería consultarse la Guía de Administración de Redes con Linux (http://es.tldp.org/Manuales-LuCAS/GARL/garl-1.0/) de Olaf Kirch, que se encuentra traducida al castellano en la web del proyecto LUCAS. (http://es.tldp.org)

Este protocolo tiene su origen en una red experimental desarrollada por el ejército de los Estados Unidos desde 1969 y que estuvo operativa en 1975 conocida como ARPANET, dando lugar a Internet en 1990 y adoptando, consecuentemente, el protocolo TCP/IP en todos sus nodos.

La novedad que aportaba esta forma de comunicación es que, hasta entonces, todos los protocolos estaban diseñados para enviar o reenviar ficheros completos, sin embargo el TCP fragmenta los datos en pequeñas unidades[3] que son enviados inmediatamente al nodo de destino que es el encargado de reensamblarlos. Esta característica supone una complejidad para el software de comunicación pero permite la ejecución de aplicaciones interactivas a través de la red. Pues bien, ya tenemos *idioma*.

2.3. Direcciones IP

Cada ordenador conectado a una red necesita estar perfectamente identificado de forma que los paquetes que lo tengan como destinatario sean capaces de localizarlo de forma inequívoca. Esta es la misión del protocolo IP.

Actualmente las direcciones IP [4] están compuestas por un número único de 32 bits que se asigna a cada nodo de la red, o más exactamente, a cada *interfaz*, normalmente la tarjeta de red. Este número suele representarse en notación decimal para cada octeto o *byte* (8 bits) con un rango de 0 a 255.

Desde los comienzos de Internet se clasificaron, tal vez arbitrariamente, las redes en diferentes tipos según el número de nodos que las componían, así tenemos:

- *Redes de clase A,* identificadas con el primer octeto de la dirección IP. Por lo tanto, pueden albergar, cada una, 16 millones de nodos, aproximadamente.
- *Redes de clase B,* identificadas con los dos primeros octetos de la dirección IP.Constan de unos 65.000 nodos cada una.
- *Redes de clase C,* identificadas con los tres primeros octetos de la dirección IP, reservando el último octeto para identificar el nodo, pudiendo estar formadas por 254 equipos.

Por lo tanto, los cuatro octetos de la dirección IP de cada ordenador, junto a la máscara de red, identifican perfectamente al equipo y a la red de la que forma parte. Esta máscara de red tiene como misión *"ocultar"* los octetos correspondientes a la identificación de la red y dejar *"visibles"* los usados para identificar el nodo.

Si las máquinas de nuestra red deben conectarse a otras redes públicamente, hemos de disponer de nuestro propio número de red asignado por el organismo regulador de la direcciones públicas de Internet, el NIC *(Network Information Center)*, mientras que si no se conectan directamente a otras redes, es decir, si no disponemos de una red con direcciones públicas, podemos asignar los números de cada nodo libremente, aunque lo más adecuado es utilizar un número de red reservado para este propósito. El rango reservado para cada tipo de red es:

Tipo de red	Máscara de red	Dirección desde	Dirección hasta
A	255.0.0.0	0.0.0.0	127.255.255.255
B	255.255.0.0	128.0.0.0	191.255.255.255
C	255.255.255.0	192.0.0.0	223.255.255.255

2.4. Construyendo nuestro ejemplo

Supongamos un centro que dispone de un aula de informática con quince ordenadores[5], dos en secretaría, uno en dirección, cinco en la sala de profesores o departamentos y un equipo dedicado a servir conforme al propósito de esta guía. Lógicamente necesitaremos una dirección de red reservada de tipo C. Por ejemplo,192.168.1.0 y todos nuestros hosts tendrán como máscara de red 255.255.255.0

Es normal reservar la dirección 192.168.1.1 para el equipo que actuará como *puerta de enlace* o *gateway* hacia el exterior, mientras que la dirección 192.168.1.255 se reserva como *broadcast* o *dirección de difusión*.[6], por lo tanto, nuestra red podría estar formada por 254 ordenadores. Todos tendían como máscara de red 255.255.255.0 que ocultaría los tres primeros octetos correspondientes a la dirección de red.

El ordenador dedicado a puerta de enlace con la Internet necesita tener instaladas dos tarjetas de red, una para comunicarse con nuestra propia intranet[7] y la otra para salir al exterior que recibirá como IP la que tenga asignada el módem-router de ADSL, para nuestros propósitos vamos a considerar que la IP 20.20.20.20 y la máscara de red 255.255.255.128 es la que corresponde a nuestro router.

2.4.1. Bautizando las máquinas

Para asignar la dirección IP a cada máquina es aconsejable seguir algún criterio identificativo, por ejemplo, a partir del 100, podemos situar los ordenadores del aula de informática, así pues, sus direcciones serían 192.168.1.101 hasta 192.168.1.115 mientras que podemos reservar las diez primeras para equipos que desempeñan tareas de servidor, como nuestra máquina que hace de puerta de enlace, un servidor de impresión, etc. Los ordenadores de los departamentos situarlos por debajo del 100 y los de secretaría por encima de 200, de esta manera sería más sencillo mantener un registro de direcciones y evitar conflictos en nuestra red.

También debemos bautizar a nuestras máquinas con nombres amigables y cariñosos, de la misma forma que sería una falta de sensibilidad llamarle a nuestro perro, *miperro*. No obstante, es aconsejable asignar los nombres de equipo con una referencia ordinal, por si en un futuro, éstos se incrementaran. Así pues con estas consideraciones, nuestra intranet local podría quedar estructurada de la siguiente forma:

Equipo	Nombre	Dirección IP	Máscara de red
Servidor	gato	192.168.1.1	255.255.255.0
Servidor	gato	20.20.20.20	255.255.255.0
Dirección	dire	192.168.1.11	255.255.255.0
Secretaría	secre1	192.168.1.201	255.255.255.0
Secretaría	secre2	192.168.1.202	255.255.255.0
Aula Informática, nº *i*	info*i*	192.168.1.10*i*	255.255.255.0

Hemos nombrado ordinalmente a los equipos de secretaría y del aula de informática, pues es de esperar -y de desear- que puedan incrementarse. Al equipo que actuará como servidor hemos optado por llamarlo **`gato`** para que nos recuerde su función como *gateway*, mientras que el de dirección lo hemos bautizado sencillamente como **`dire`**.

2.4.2. Preparando nuestro servidor

Empezaremos por activar las dos tarjetas que hemos instalado en nuestro servidor, es decir, vamos a configurar las interfaces de red para que cada una tenga la dirección IP adecuada y sea capaz de cumplir con su misión.

En linux, las interfaces de red tienen un nombre secuencial `eth0,eth1,...` según la cantidad de tarjetas ethernet que tengamos instaladas. En nuestro caso, la primera tarjeta (`eth0`) la destinaremos a las comunicaciones con nuestra intranet y le asignaremos, por lo tanto, la IP 192.168.1.1, mientras que la segunda (`eth1`) la dedicaremos a comunicarse con Internet y le asociaremos la IP 20.20.20.20 que hemos convenido como dirección pública del router.

```
[root@gato root]# ifconfig eth0 192.168.1.1 netmask 255.255.255.0 up
[root@gato root]# ifconfig eth1 20.20.20.20 netmask 255.255.255.128 up
```

Esto activaría nuestras tarjetas de red. Si queremos comprobar que todo ha funcionado podemos consultar la configuración ejecutando:

```
[root@gato root]# ifconfig -a
```

Obteniendo una información con estas características:

```
eth0      Link encap:Ethernet  HWaddr 00:04:76:EA:A3:61
          inet addr:192.168.1.1  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:9 Base address:0xd400

eth1      Link encap:Ethernet  HWaddr 00:04:76:FA:B3:61
          inet addr:20.20.20.20  Mask:255.255.255.128
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:10 Base address:0xd200

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:244 errors:0 dropped:0 overruns:0 frame:0
          TX packets:244 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:15380 (15.0 Kb)  TX bytes:15380 (15.0 Kb)
```

Los dos primeros bloques corresponden a la información relativa a nuestras dos tarjetas de red mientras que el último refleja la interfaz de loopback, siempre presente en todo sistema linux aunque no se posea ningún dispositivo de red.

Si no disponemos de una conexión a Internet mediante ADSL, sino que lo hacemos a través de un módem, todo lo dicho para la tarjeta de red `eth1` quedaría sin efecto y en su lugar deberíamos ver cuando estamos conectados:

```
eth0      Link encap:Ethernet  HWaddr 00:04:76:EA:A3:61
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:9 Base address:0xd400

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:264 errors:0 dropped:0 overruns:0 frame:0
          TX packets:264 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:16678 (16.2 Kb)  TX bytes:16678 (16.2 Kb)

ppp0      Link encap:Point-to-Point Protocol
          inet addr:62.201.10.34  P-t-P:62.201.10.5  Mask:255.255.255.255
          UP POINTOPOINT RUNNING NOARP MULTICAST  MTU:1500  Metric:1
          RX packets:25 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:3
          RX bytes:7943 (7.7 Kb)  TX bytes:1612 (1.5 Kb)
```

2.5. Resolución de nombres

Cuando enviamos un paquete de datos a un ordenador, es necesario que llegue a su destino. Si usamos su dirección IP no presenta mayor problema, sin embargo los números de 32 bits usados para identificar un equipo no son fáciles de recordar mientras que los ordenadores, como hemos visto en la Sección 2.4.1, suelen tener nombres más intuitivos

para el ser humano. La aplicación que se encarga de traducir nombres de máquina a direcciones IP se conoce como *Sistema de Resolución de Nombres* o DNS *(Domain Name System)*.

En una red pequeña no resultaría difícil mantener una tabla de resolución de nombres almacenada en el fichero `/etc/hosts` de cada máquina que asociara a cada dirección IP el nombre de esa máquina, sin embargo en toda Internet, fácilmente se comprende que no sería una buena solución, por ello en 1984 se diseñó y adoptó un sistema nuevo también llamado DNS que consiste básicamente en dividir los nombres de máquina en zonas o dominios y delegar en unos servidores de nombres que mantengan toda la información acerca de una zona. Por ejemplo, cuando cursamos una petición o enviamos un paquete de datos para la máquina **`hispalinux.es`** el sistema de resolución de nombres interroga a un servidor que mantenga la zona de España (.es) y éste le indicará cúal es la dirección IP que corresponde al ordenador con ese nombre.

Bien, dicho así parece sencillo, aunque en realidad es un sistema bastante complejo y con muchos matices pero que escapan totalmente de nuestros propósitos.

2.5.1. DNS en nuestras máquinas

Todos nuestros equipos van a necesitar resolver nombres de máquinas, tanto para navegar por internet como, tal vez, localmente en nuestra propia red para algunos servicios que tengamos implementados en las máquinas locales.

En el caso local, el fichero responsable de la resolución de nombres es `etc/hosts` y su estructura es simplemente una tabla conteniendo en cada entrada la dirección IP y el nombre de la máquina. Obviamente no es necesario que es este fichero estén identificados todos los equipos de nuestra intranet, sino solamente aquellos a los que debamos dirigirnos con algún propósito, amén del propio ordenador. Para configurarlo bastará usar cualquier editor de textos y dejarlo de la siguiente manera:

```
# Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1 dire localhost.localdomain localhost
192.168.1.11 dire
192.168.1.1     gato
```

Este podría ser el fichero `etc/hosts` para el ordenador de dirección.

Cuando resolvemos nombres de máquinas en Internet ya no usaremos este fichero[8] sino que ahora, el responsable es el archivo `etc/resolv.conf` que contendrá las direcciones de nuestros servidores de nombres (DNS) proporcionados por nuestro proveedor de acceso[9] . De nuevo, usando nuestro editor favorito dejaremos el archivo `etc/resolv.conf` de esta guisa:

```
nameserver 150.214.3.9
nameserver 150.214.90.11
nameserver 212.59.199.2
```

Podemos colocar hasta tres servidores de nombres, en previsión de que alguno de ellos se caiga en un momento dado.

Para aclarar, un poco, cómo funciona este asunto digamos que cuando una de nuestras máquinas necesita resolver un nombre, mira en primer lugar en el fichero `etc/hosts` y finalmente, si no consigue resolverlo, interroga a las máquinas especificadas en el `etc/resolv.conf` puesto que así se lo indica el fichero de configuración `/etc/nsswitch.conf` donde se le especifican los recursos de resolución de nombres en un línea que luce:

```
hosts:  files nisplus nis dns
```

Esta línea instruye a la computadora que, para resolver un nombre debe mirar en `etc/hosts` en primer lugar, según el parámetro **`files`**, después debe usar **`nisplus`** o **`nis`** caso de que tengamos implementados estos servicios en nuestra red local, y finalmente debe usar el sistema **`dns`** de los servidores de nombres.

Notas

1. Para ser precisos deberíamos decir que una red es un conjunto de *nodos* ya que éstos no tienen que ser necesariamente ordenadores tal y como los concebimos, pueden ser -de hecho así ocurre en muchas ocasiones- simplemente terminales sin disco o impresoras inteligentes.
2. Protocolo de Control de la Transmisión *(Transmission Control Protocol)*/Protocolo de Internet *(Internet Protocol)*
3. paquetes
4. La versión actual es IPv4 aunque parece ser que se está preparando la IPv6 donde cada equipo tendrá un número de 128 bits, aumentando exponencialmente las posibilidades de expansión de la red Internet.
5. También llamados *hosts*, en este contexto.
6. Esta *dirección de difusión* se usa para enviar paquetes de datos que todos los equipos de la red deben recibir.
7. Que será la que tenga la dirección IP del gateway 192.168.1.1
8. Sería un disparate que nuestro fichero `etc/hosts` almacenase todas las IP públicas de Internet
9. En realidad no es imprescindible usar los DNS de nuestro acceso a Internet, funciona con cualquier servidor de nombres público.

Capítulo 3. Enmascaramiento

3.1. Sobre los puertos

El conocimiento y la gestión de los puertos no es un tema baladí, sin embargo, aún a riesgo de ser poco rigurosos podríamos compararlos con las cadenas de TV que recibimos en nuestros televisores, cada una la recepcionamos por un canal diferente en el mismo aparato y con el mismo cable.

Algo parecido ocurre. Cuando un ordenador envía una petición a otro equipo de la red, ¿cómo sabe éste qué tipo de servicio es solicitado?, simplemente porque la petición entra por un puerto determinado, es decir, existe una correspondencia común entre puertos y servicios que podemos ver ejecutando:

```
$cat /etc/services
```

Obteniendo un listado de todos los servicios y puertos disponibles del tipo:

```
ftp             21/tcp
ftp             21/udp          fsp fspd
ssh             22/tcp                          # SSH Remote Login Protocol
ssh             22/udp                          # SSH Remote Login Protocol
telnet          23/tcp
telnet          23/udp
# 24 - private mail system
smtp            25/tcp          mail
smtp            25/udp          mail
time            37/tcp          timserver
time            37/udp          timserver
.........................................
http            80/tcp          www www-http    # WorldWideWeb HTTP
http            80/udp          www www-http    # HyperText Transfer Protocol
kerberos        88/tcp          kerberos5 krb5  # Kerberos v5
kerberos        88/udp          kerberos5 krb5  # Kerberos v5
```

Donde se observa el servicio, el puerto/protocolo y una descripción del servicio prestado.Por eso, cuando solicitamos una página web, salvo que indiquemos expresamente otra cosa, estaremos llamando al puerto 80 del servidor solicitado. Una vez recibida nuestra petición, si es razonable, se iniciará un pequeño diálogo con nuestra computadora para que se realice la conexión y, finalmente, veremos aparecer en nuestro navegador la página solicitada. Esta situación acontece cuando navegamos por internet con nuestra conexión facilitada por un proveedor de ISP. Al conectarnos, nuestro proveedor nos asigna una dirección IP pública que, normalmente varía de una conexión a otra.[1] Esta dirección IP asignada a nuestro equipo por el proveedor de ISP es la que permite a cada servidor web remitirnos las páginas que le solicitamos.

3.2. El sistema NAT.Enmascaramiento y filtrado de direcciones IP

Todo el proceso descrito en la Sección 3.1ocurre al conectarnos de manera sencilla con nuestro ordenador, un módem y una cuenta de acceso a Internet. Ahora bien, cuando tenemos una red y deseamos navegar desde cualquier equipo de la misma, la cuestión es diferente.

Imaginemos que el ordenador nº 3 del aula de informática, **`info3`**, con dirección IP 192.168.1.103 quiere consultar una documentación sobre linux en castellano y, para ello, realiza una petición al puerto 80 de la máquina **`es.tldp.org`** tecleando en su navegador **`http://es.tldp.org`**

Cuando el servidor de nombres le proporcione la IP adecuada y descubra que no pertenece a su red local, encaminará su petición a través de **`gato`** y éste redirigirá la petición al servidor **`es.tldp.org`** que la recibirá procedente de la IP 20.20.20.20 y, en consecuencia, le responderá a **`gato`** que tiene que saber, de alguna manera servirle esta información a **`info3`** que fue, en realidad, quien solicitó esta página. Para ello, **`gato`** marca el paquete que recibe de **`info3`** asignándole un dato más que identifica el remitente de la petición, de modo que, al recibir la respuesta, sepa que no es para ella y redirigirla a quien la solicitó, en este caso **`info3`**.

Todo este proceso que realiza **`gato`** se conoce con el nombre de *enmascaramiento* o *masquerading* y es una característica del núcleo de linux, a partir de su versión 2.0 que permite la navegación a toda una red local con una sola conexión a internet sin necesidad de usar un proxy. Es una de las posibilidades que nos ofrece el sistema NAT (*Network Address Translation*) o *Traducción de Direcciones de Red.*

Puede ampliarse la información relativa al funcionamiento de NAT en el siguiente documento (http://www.netfilter.org/documentation/HOWTO/es/NAT-HOWTO.txt) en castellano, aunque los sitios oficiales del proyecto NAT se encuentran en Penguin Computing (http://netfilter.filewatcher.org/), el equipo Samba y SGI (http://netfilter.samba.org/) y, por último, el mantenido por Harald Welte (http://netfilter.gnumonks.org/).

3.2.1. Filtrado de paquetes

Una red nunca es segura. Esta es la principal idea que nunca debe abandonar un administrador de red. Por lo tanto, con esa premisa, si permitimos el tráfico en nuestra red debemos tomar todas las precauciones a nuestro alcance para que impedir que los paquetes con código malicioso o no deseado circulen y que nos ocasionen algún quebradero de cabeza.

Un filtro es, pues, un software que examina las cabeceras de los paquetes y decide su suerte conforme a unas reglas fijadas por el usuario. Este software está situado en el propio núcleo de Linux y se conoce con el nombre de **`netfilter`**, mientras que la herramienta encargada de su gestión, **`iptables`** es la que se comunica con el núcleo y le proporciona las reglas de filtrado de los paquetes. Básicamente son dos las acciones para con los paquetes examinados: DROP (Descartarlo y hacer como si no se hubiera recibido, con lo que el paquete se pierde irremediablemente) o ACCEPT (Dejar que pase y alcance su destino), aunque el abanico de posibilidades de uso de esta herramienta es tan sumamente amplio y poderoso que la creación de un buen cortafuegos es objeto de sesudos tratados y aquellos capaces de construirlos con robustez, son muy pocos y muy bien pagados.

Aquellas mentes inquietas que quieran sumergirse en este apasionante mundo de la seguridad en las redes pueden comenzar con este documento (http://www.netfilter.org/documentation/HOWTO/es/packet-filtering-HOWTO.txt) en castellano. Los sitios oficiales son los mismos que para el enmascaramiento, ya apuntados en la Sección 3.2.

> **Importante:** La utilidad **`iptables`** añade y elimina reglas de la tabla de filtrado del núcleo, es decir, cualquier política que establezcamos con ella se perderá al reiniciar el sistema, a menos que las establezcamos como "permanentes" para que Linux las reconozca en el siguiente arranque.

3.2.2. Enmascarando y filtrando con `gato`

Con Linux ha sido posible el enmascaramiento desde los arcaicos núcleos de la serie 1.1, aunque las herramientas han evolucionado y a partir de las versión 2.4 del núcleo se utiliza **`iptables`** para controlar todo el proceso.

Es normal que todo pueda resultar algo farragoso e incluso demasiado complicado, pero para nuestro objetivo, trataremos de simplificar al máximo y conseguiremos de una forma sencilla que nuestro equipo sea capaz de enmascarar nuestra red local para navegar por internet.

En primer lugar hemos de hacer que **`gato`** tenga habilitado el reenvio de paquetes, esto se consigue con la orden:

```
[root@gato root]#  echo 1 > /proc/sys/net/ipv4/ip_forward
```

Esta instrucción escribe el archivo `/proc/sys/net/ipv4/ip_forward` que es un fichero de texto que sólo contiene el carácter "1".

Ahora nos disponemos a limpiar todas las reglas de filtrado que pudieran existir en nuestra máquina, para ello ejecutamos:

```
[root@gato root]# iptables --flush
[root@gato root]# iptables --table nat --flush
[root@gato root]# iptables --delete-chain
[root@gato root]# iptables --table nat --delete-chain
```

Y, por último vamos a enmascarar nuestra red local y permitir que navegue con las dos instrucciones siguientes:

```
[root@gato root]# iptables --table nat --append POSTROUTING --out-interface eth1 -j MASQUERADE
[root@gato root]# iptables --append FORWARD --in-interface eth0 -j ACCEPT
```

Se observa que las órdenes son bastante intuitivas. En la primera estamos diciendo que enmascare y enrute los paquetes a través de la segunda tarjeta de red que ya configuramos para comunicarnos con Internet. En la segunda, establecemos la política de aceptar los paquetes que reciba por la primera tarjeta de red (la que comunica con la red local) y los reenvíe.

Hemos de aclarar que todo este montaje también sirve si queremos sacar nuestra red local a Internet aunque no dispongamos de ADSL sino de un módem convencional. El único cambio sería sustituir la eth1, que es la tarjeta dedicada a conectar con el router, por ppp0 que es la interfaz del módem. Es decir, en caso de disponer de un módem para conectar a Internet, las instrucciones serían:

```
[root@gato root]# iptables --table nat --append POSTROUTING --out-interface ppp0 -j MASQUERADE
[root@gato root]# iptables --append FORWARD --in-interface eth0 -j ACCEPT
```

Llegado a este punto, ya estamos en condiciones de comprobar que cualquier ordenador de nuestra red local es capaz de comunicarse con cualquier equipo externo que posea una IP pública. Podríamos comprobarlo usando simplemente:

```
[usuario@info5 usuario]$ ping www.um.es
```

Deberíamos ver

```
PING araneus.um.es (155.54.1.244) from 62.201.10.34 : 56(84) bytes of data.
64 bytes from araneus.um.es (155.54.1.244): icmp_seq=1 ttl=245 time=220 ms
64 bytes from araneus.um.es (155.54.1.244): icmp_seq=2 ttl=245 time=206 ms
64 bytes from araneus.um.es (155.54.1.244): icmp_seq=3 ttl=245 time=210 ms
```

```
64 bytes from araneus.um.es (155.54.1.244): icmp_seq=4 ttl=245 time=197 ms

--- araneus.um.es ping statistics ---
5 packets transmitted, 4 received, 20% loss, time 4007ms
rtt min/avg/max/mdev = 197.133/208.841/220.496/8.386 ms
```

Esto probaría, no sólo que el ordenador número de cinco del aula de informática ha sido capaz de comunicarse con el servidor de la Universidad de Murcia, sino también, que funciona la resolución de nombres, por lo que navegaríamos ya sin problemas. Además, descubrimos que la máquina que sirve web se llama **`araneus`** con IP pública 155.54.1.244 y que nuestro proveedor de Internet nos ha asignado la IP 62.201.10.34 para esta sesión.

Notas

1. salvo que deseemos pagar más y adquirir nosotros nuestra IP pública propia.

Capítulo 4. Sirviendo web

4.1. Introducción

Uno de los objetivos de este montaje es aprovechar nuestro servidor para que proporcione, tanto al exterior como a la intranet, servicios de Internet. Uno de ellos, tal vez el más popular, el servidor web.

El propósito de este tipo de software es transmitir páginas en formato `html` a los navegadores cliente que las soliciten, utilizando para ello el protocolo de comunicación `http`, aunque actualmente, es capaz de servir más tipos de contenidos, bien por sus propios medios o a través de la interacción con otros programas.

4.2. El servidor web Apache

Hoy en día es el servidor web más utilizado del mundo, encontrándose muy por encima de sus competidores, tanto gratuitos como comerciales. Es un software de código abierto[1] que funciona sobre cualquier plataforma. Por supuesto, se distribuye prácticamente con todas las implementaciones de Linux.

Tiene capacidad para servir páginas tanto de contenido estático, para lo que nos serviría sencillamente un viejo ordenador 486, como de contenido dinámico a través de otras herramientas soportadas que facilitan la actualización de los contenidos mediante bases de datos, ficheros u otras fuentes de información.

4.2.1. Instalación

En caso de que no instalaramos desde el principio este servidor, siempre podremos instalarlo después de una forma cómoda y sencilla con la tecnología de *paquetes* autoinstalables que poseen tanto Red Hat como Debian, dos de las implementaciones más usuales de Linux.

En el caso de Red Hat 8.0, por ejemplo, los paquetes son:

- apacheconf-0.8.1-1.rpm
- apache-1.3.22-2.rpm
- apache-manual-1.3.22-2.rpm
- apache-devel-1.3.22-2.rpm

El único paquete realmente imprescindible es el segundo, puesto que el primero permite una configuración en modo gráfico, el tercero contiene toda la documentación y el último es el paquete de desarrollo donde se incluyen las fuentes del programa.

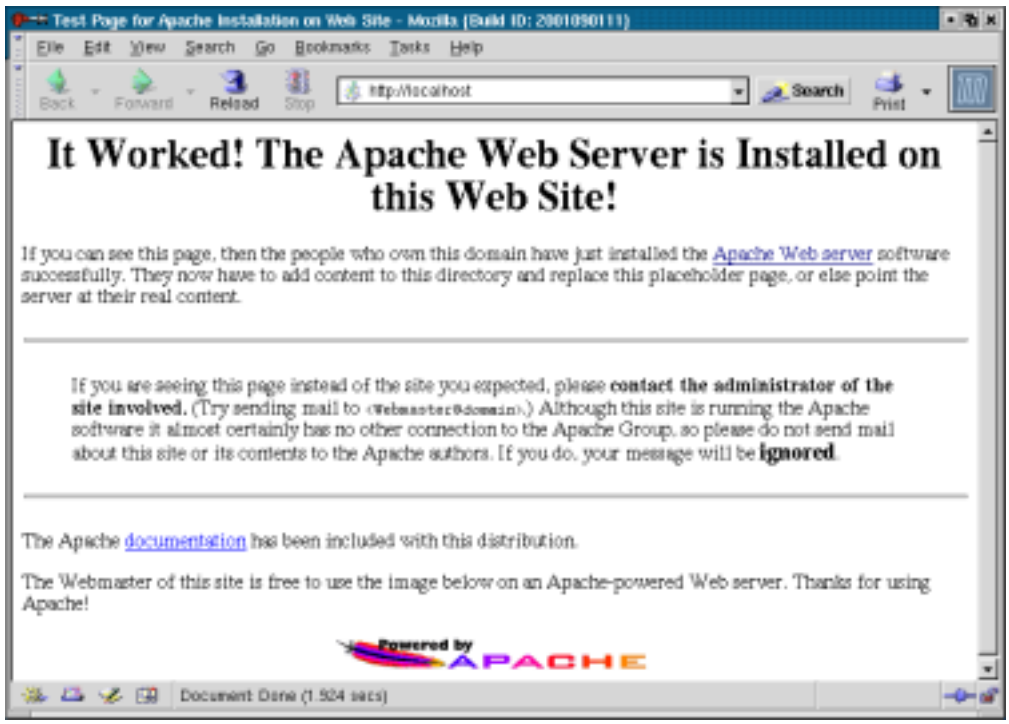
Una vez instalado, se comporta como servidor independiente y viene preparado por defecto para atender peticiones realizadas a través del puerto 80 de nuestro equipo. Podemos interactuar con él de forma manual empleando las órdenes:

```
[root@gato root]# service httpd start
[root@gato root]# service httpd stop
[root@gato root]# service httpd restart
[root@gato root]# service httpd status
```

Si deseamos arrancarlo, pararlo, reiniciarlo o comprobar su estado, respectivamente.

4.2.2. Configuración

Una vez realizada la instalación, Apache queda listo para trabajar. Podemos comprobar que funciona tecleando en nuestro navegador favorito **`http://192.168.1.1`** se nos mostrará entonces una pantalla de bienvenida del servidor (*It worked!*), síntoma de que hemos contactado con él y se encuentra funcionando.



Pantalla de bienvenida al servidor Apache

Notas

1. Es un software de libre distribución que publica su código fuente, lo que permite que cualquiera pueda modificarlo y colaborar así a su desarrollo.

Capítulo 5. Sirviendo ficheros

5.1. Protocolo de transferencia de ficheros. FTP

Al fin y al cabo, todos los conceptos aquí destilados tienen sentido porque estamos situados en una red. Por lo tanto tenemos como objetivo, en este ambiente, la transmisión de información de todo tipo. En el Capítulo 4 hemos utilizado el protocolo http[1] para compartir información de tipo html, principalmente, a través de nuestro servidor web *Apache.*

Continuando con esta idea vamos a continuar instalando otro servidor que será capaz de facilitar sus *clientes* cualquier tipo de fichero, a la vez que permitirá que determinados usuarios puedan escribir en ciertas zonas del disco de nuestro servidor. Para ello necesitaremos un software que controle todo el proceso (ProFTPD) y un protocolo de comunicación (FTP, File Transfer Protocol), Protocolo de *Transferencia de Ficheros*.

5.1.1. Características de FTP

El protocolo FTP permite la transferencia de ficheros de un ordenador a otro, bajo un control ejercido por el software servidor y por la configuración de éste que hayamos fijado.

Cuando se establece una comunicación mediante FTP entre dos máquinas ha de superarse una fase previa de autentificación basada en un fichero de contraseñas, de la misma forma que se nos proporciona una shell en un ordenador, bien de modo local o remotamente a través de telnet. Sin embargo, esta comunicación establecida no es segura, tampoco tiene capacidades de filtrado aunque resultaría muy difícil tomar el control de una máquina, sólo con la conexión vía FTP.

Pese a todo, constituye una de las herramientas más útiles para el intercambio de ficheros entre diferentes ordenadores y es la forma habitual de publicación en Internet.

Aunque puedan contemplarse otras posibilidades, hay dos tipos fundamentales de acceso a través de FTP:

- *Acceso anónimo*, cuando el contacto con lamáquina lo realiza un usuario sin autentificar y sin ningún tipo de privilegio en el servidor. En ese caso, el usuario es confinado a un directorio público donde se le permite descargar los archivos allí ubicados pero sin posibilidad de escribir ningún fichero. No se le permite, normalmente, subir de nivel y listar los contenidos de los directorios de nivel superior.
- *Acceso autorizdo*, cuando el usuario que solicita la conexión tiene una cuenta con ciertos privilegios en el servidor y, tras autentificarse, se le confina a su directorio predeterminado desde donde puede descargar ficheros y, si la política del sistema se lo permite, también escribir, aunque normalmente se limita su espacio mediante una cuota de disco. Puede estar autorizado a recorrer parte del árbol de directorios y listar su contenido o escribir el ellos, dependiendo del tipo de privilegios que posea.

5.2. El servidor ProFTPD

Tradicionalmente el servidor FTP que solían suministrar la mayoría de distribuciones era **`wu.ftpd`**, sin embargo presentaba algunos agujeros de seguridad y no respondía a las espectativas de un servidor de ficheros en la actualidad. Fue concebido de una forma más doméstica, para pequeños intercambios en redes de confianza y se ha visto desbordado por el auge de Internet y sus políticas de hospedaje.

El servidor **ProFTPD** ha sido creado para responder, desde el software libre a estas limitaciones. Pese a su relativa juventud presenta un diseño mucho más robusto y más simple que le permiten ofrecer unas prestaciones que con **wu.ftpd** resultaban muy engorrosas e inseguras.

Suele venir incluido en la mayoría de distribuciones y, por lo tanto, puede ser instalado durante el proceso de instalación inicial, de no ser así, la forma más cómoda en localizar los dos paquetes[2] en que suele ser distribuido y proceder a su instalación, decidiendo que queremos que arranque de forma autónoma (standalone) o bien a través de `inetd`.

5.3. Configuración de ProFTPD

El fichero de configuración que controla todo el proceso, normalmente se encuentra situado en `/etc/proftpd.conf` y posee una estructura de instrucciones que deben considerarse imbricadas, es decir solapadas, permitiendo la concreción de la configuración directorio a directorio, aunque algunos comandos se apliquen a la totalidad del servidor.

En cualquier caso, la estructura del fichero es similar a un documento etiquetado de tipo HTML, donde la directiva irá expresada entre las etiquetas correspondientes. Por ejemplo:

```
<Limit LOGIN>
DenyUser paco,pepa #Impide el acceso a paco y pepa
</Limit>
```

La directiva anterior denegaría el acceso al servidor FTP a los usuarios **paco** y **pepa** mientras que la frase precedida por la almohadilla "#" responde a la buena costumbre de comentar los ficheros de configuración.

En nuestro caso, el fichero de configuración es el siguiente:

```
# This is the ProFTPD configuration file

ServerIdent   on "Servidor FTP preparado."
ServerAdmin   cvalera@larural.es
ServerType   standalone
#ServerType   inetd
DefaultServer   on
AccessGrantMsg   "Usuario %u registrado."
DisplayConnect   /etc/ftpissue
#DisplayLogin   /etc/ftpmotd
#DisplayGoAway   /etc/ftpgoaway

# Use pam to authenticate by default
AuthPAMAuthoritative  on

# Do not perform ident lookups (hangs when the port is filtered)
IdentLookups   off

# Port 21 is the standard FTP port.
Port    21

# Umask 022 is a good standard umask to prevent new dirs and files
# from being group and world writable.
Umask    022

# Chmod isn't allowed by default
```

```
AllowChmod   on

# Default to show dot files in directory listings
#LsDefaultOptions  "-a"

# See Configuration.html for these (here are the default values)
#MultilineRFC2228  off
#RootLogin   off
#LoginPasswordPrompt  on
#MaxLoginAttempts  3
#MaxClientsPerHost  none

# To prevent DoS attacks, set the maximum number of child processes
# to 30.  If you need to allow more than 30 concurrent connections
# at once, simply increase this value.  Note that this ONLY works
# in standalone mode, in inetd mode you should use an inetd server
# that allows you to limit maximum number of processes per service
# (such as xinetd)
MaxInstances   30
MaxClientsPerHost  1
DisplayLogin  .welcome.msg
DisplayQuit  .quit.msg
# Set the user and group that the server normally runs at.
User    nobody
Group    nobody

# Normally, we want files to be overwriteable.
<Directory /*>
  AllowOverwrite  on
</Directory>
DefaultRoot ~
# A basic anonymous configuration, no upload directories.
<Anonymous ~ftp>
  # Uncomment the following line to allow anonymous access
  RequireValidShell  off
  AllowChmod   off

  User    ftp
  Group    ftp
 #AccessGrantMsg  "Acceso anónimo correcto. Restringido"

  # We want clients to be able to login with "anonymous" as well as "ftp"
  UserAlias   anonymous ftp

  # Limit the maximum number of anonymous logins
  MaxClients   10

  # We want 'welcome.msg' displayed at login, '.message' displayed in
  # each newly chdired directory and tell users to read README* files.
  DisplayLogin   .welcome.msg
  DisplayFirstChdir  .message
  DisplayReadme   README*
```

```
  # Limit WRITE everywhere in the anonymous chroot
  <Limit WRITE>
    DenyAll
  </Limit>

</Anonymous>
```

Notas

1. Hipertext transfer protocol *(Protocolo de transferencia de hipertexto)*
2. En http://rpmfind.net, por ejemplo, si estamos usando una distribución que usa el sistema de empaquetado de *rpm* como Red Hat.

Capítulo 6. El servidor proxy

Prodríamos traducir la palabra *proxy* por, algo así como, *"hacer algo en nombre de otro."* En el contexto en que nos movemos, un servidor proxy se encarga de actuar en nombre de muchos clientes, en concreto, trabaja para todas las máquinas de nuestra red que tengamos autorizadas para navegar por Internet.

Un proxy HTTP es una máquina (**`gato`**)que recibe peticiones de páginas web de otra máquina (**`info7`**, por ejemplo). El proxy, negocia esta petición, a su vez, con el servidor web adecuado, obtiene la página solicitada y retorna el resultado a **`info7`**.

Otra característica muy importante en un servidor proxy es que puede tener un caché[1] con las páginas recibidas, de modo que si otra máquina solicitase una dirección ya visitada con anterioridad por cualquier máquina de la red, le enviará la copia local que reside en el caché. Eso permite un uso eficiente del ancho de banda y un menor tiempo de respuesta.

Es decir, supongamos que a primera hora de la mañana, una de las máquinas de secretaría realiza una petición para consultar el último BOJA. Más tarde, el director decide también realizar la misma petición. Pues bien, el servidor proxy le facilitará a la máquina de dirección la copia local del caché de la página del BOJA almacenada con anterioridad. Evidentemente, esta situación es beneficiosa, tanto para la velocidad de respuesta ante nuestra intranet, como para no saturar innecesariamente el ancho de banda de toda la Internet.

Además, puesto que las máquinas cliente no están directamente conectadas al exterior, ésta es una forma de incrementar la seguridad de la red interna. Un proxy bien configurado puede ser tan efectivo como un buen firewall.

Existen muchos servidores proxy para Linux. Una solución muy extendida es utilizar el propio modulo de proxy del servidor web Apache que hemos instalado, sin embargo, la elección más completa y robusta es la instalación del proxy SQUID, también de libre distribución.

6.1. Consideraciones sobre la caché

Para una gestión correcta e inteligente del uso de la caché, es necesario tener en cuenta con mucha claridad, qué objetos deben ser cacheados. Por ejemplo, es totalmente inapropiado cachear los números de tarjetas de crédito, los resultados de un script ejecutado remotamente, sitios que cambian frecuentemente e incluso sitios que no desean ser cacheados.

Los scripts ejecutables cgi-bin normalmente no son almacenados en la caché, como tampoco lo son las páginas que indican en sus cabeceras periodos de caducidad, Así que es posible especificar con reglas extra qué se debe, qué no se debe cachear, y por cuánto tiempo.

Si queremos determinar la utilidad y rendimiento de la caché, debemos tener en cuenta que con una cache pequeña (un par de gigas) se obtienen unos resultados altos (cercanos al 25%). Pero doblando el espacio en disco, no necesariamente se dobla este porcentaje, pueto que se intenta capturar unas peticiones, que con frecuencia son poco utilizadas. Una caché grande (por encima de 20 Gb) probablemente no llegará al 50%, a no ser que las páginas se mantengan durante mucho tiempo.

6.2. El proxy SQUID

SQUID es un software de libre distribución para realizar la tarea de un servidor proxy con prestaciones muy profesionales. Suele acompañar a las distribuiciones más habituales, aunque también puede obtenerse de su sitio oficial (http://www.squid-cache.org/), pero lo más sencillo es utilizar el empaquetado en formato `rpm`, si estamos usando una

distribución Red Hat o `deb` si trabajamos con Debian, en caso de que usted no domine la instalación a partir de las fuentes, aunque hemos de decir que la mayoría de software viene con sencillos scripts de instalación que le facilitan la tarea a los usuarios menos avezados.

6.2.1. Instalación

Hemos de obtener, en nuestro caso, el paquete de una versión igual o superior a `2.4.STABLE1`, cualquier versión anterior no es recomendable ya que presenta ciertas carencias de seguridad, por lo tanto, en todos los casos asegúrese de disponer de la última versión estable del software que piense instalar. Una vez obtenido bastará teclear:

```
[root@gato root]#rpm -ivh squid-2.4.STABLE1.i386.rpm
```

Esta instrucción hará que SQUID se instale en su sistema.

6.3. Configuración de SQUID

El fichero de configuración de SQUID se halla en `/etc/squid/squid.conf` y hemos de editarlo con nuestra herramienta favorita para realizar los cambios adecuados y conseguir que cumpla su tarea con cierta seguridad para nuestro sistema.

Este fichero de configuración consta de multitud de parámetros configurables que ajustan el servidor a nuestras necesidades. Trataremos de reflejar aquellos indispensables para un óptimo funcionamiento.

6.3.1. Puerto para SQUID

Por defecto, SQUID utilizará el puerto 3128, vea Sección 3.1 para una visión más general acerca de los puertos y su misión, aunque puede configurarse para que use cualquier otro, incluso varios puertos simultáneamente, dependiendo de nuestras necesidades. La línea correspondiente quedará:

```
http_port 3128
```

6.3.2. Tamaño de caché

En esta instrucción fijamos el espacio en disco que se usará para almacenar las páginas visitadas, es decir, respondemos a la pregunta, Â¿Cuánto deseo almacenar de Internet en mi disco duro?

Por defecto SQUID usará 100 Mb, como límite para el tamaño del caché, pero si quisiéramos fijar, por ejemplo 500 Mb, debemos fijar la entrada correspondiente de la siguiente forma:

```
cache_dir ufs /usr/local/squid/cache 500 16 256
```

En la línea anterior estamos seleccionando el directorio de caché (`/usr/local/squid/cache`), indicando el tamaño máximo para éste (500), la cantidad de subdirectorios de primer nivel que puede contener (16, el valor por defecto) y, el número de subdirectorios de segundo nivel (256, también por defecto) que puede almacenar

> **Importante:** En caso de que fijemos un tamaño para el directorio de caché superior a la capacidad real del disco duro, SQUID se bloqueará.

6.3.3. Vida en el caché

Podemos configurar también el tiempo que pueden permanecer los objetos almacenados en el caché, dependiendo de nuestras necesidades, lógicamente. De modo general, si definimos un tiempo de permanencia demasiado bajo, estaremos desaprovechando una de las principales ventajas del uso de servidor proxy, mientras que si establecemos un periodo demasiado alto, también saturaremos innecesariamente la capacidad de almacenaje.

Parece una decisión razonable, en la mayoría de casos, fijar un mes de vida para los objetos del caché. Esto se logra con la instrucción:

```
reference_age 1 month
```

6.3.4. Controles de acceso

Una de las características más interesantes de este servidor proxy es la posibilidad de establecer unas reglas de control de acceso que pueden complementar perfectamente nuestro objetivo de filtrado de paquetes.

Para ello, confeccionaremos unas *Listas de Control de Acceso* para designar qué máquinas o redes tienen permitido, o no, acceder al servidor. Cada una de ellas tendrá asociada unas *Reglas de Control* que regulará esta actividad. Es decir, definimos unas listas, por una parte y establecemos unas reglas específicas para cada una de ellas. Veamoslo con un ejemplo:

Si queremos permitir el acceso al proxy para todas las máquinas de nuestra red, hemos de definir una lista que identifique a toda nuestra red local, en nuestro caso:

```
 acl nuestrared src 192.168.1.0/255.255.255.0
```

De esta forma nuestra red queda identificada para SQUID, quedando la sección de *Listas de Control de Acceso* con el siguiente aspecto:

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl nuestrared src 192.168.1.0/255.255.255.0
```

Ahora debemos permitir el acceso con una línea con la siguiente sintaxis:

```
http_access allow todalared
```

De modo que la sección *Reglas de Control*, debe quedar:

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow nuestrared
http_access deny all
```

La línea `http_access allow nuestrared` permite el acceso al proxy para la lista denominada `nuestrared` que está formada por `192.168.1.0/255.255.255.0`, o sea, cualquier máquina cuya dirección IP esté comprendida entre `192.168.1.1` y `192.168.1.254`

Evidentemente si la línea fuese `http_access deny nuestrared` no se permitiría el acceso a ninguna máquina de nuestra red local.

Veamos un ejemplo más complejo que nos mostrará las capacidades de filtrado de SQUID. Para ello, vamos a suponer que deseamos permitir el acceso a toda nuestra red, excepto a las máquinas del aula de informática. Vamos a crear un fichero de texto que contenga las direcciones IP de éstas máquinas al que llamaremos y ubicaremos en `/etc/squid/informatica`

```
192.168.1.101
192.168.1.102
192.168.1.103
192.168.1.104
192.168.1.105
192.168.1.106
192.168.1.107
192.168.1.108
192.168.1.109
192.168.1.110
192.168.1.111
192.168.1.112
192.168.1.113
192.168.1.114
192.168.1.115
```

Le vamos a llamar `info` a esta lista que hemos creado utilizando el fichero anterior.

```
acl info src "/etc/squid/informatica"
```

De modo que ahora nuestra sección de listas quedaría:

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl nuestrared src 192.168.1.0/255.255.255.0
acl info src "/etc/squid/informatica"
```

Y, finalmente, para conseguir nuestro objetivo, la sección de reglas quedaría:

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow nuestrared !info
http_access deny all
```

El secreto está en la línea `http_access allow nuestrared !info` donde estamos permitiendo el acceso para la lista `nuestrared`, o sea toda la red, excepto a las máquinas definidas en la lista `info`, es decir, las indicadas en el fichero `/etc/squid/informatica` que son las del aula de informática, tal y como pretendíamos.

6.3.5. Redireccionamiento a través de SQUID

Una posibiliad que nos interesa plantearnos es utilizar nuestro servidor proxy de modo "transparente", es decir, nuestras máquinas saldrán a la red Internet a través de él, pero de la misma forma que lo harían mediante el router ADSL. Para ello, usando *iptables* redireccionaremos todas las peticiones al puerto 80 de nuestra red local hasta el puerto 3128 donde escucha SQUID, la instrucción para conseguirlo es:

```
[root@gato root]#/sbin/iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

Análogamente podremos redirigir otros servicios, tal y como, se reflejó en la Sección 3.2

Notas

1. Va almacenando las páginas recibidas en un directorio, igual que se comporta en caché de cualquier navegador.

Capítulo 7. Ajustes en el router ADSL

7.1. ADSL y Linux

Actualmente la conexión a Internet mediante ADSL constituye prácticamente la única solución de banda ancha para los usuarios alejados de los grandes núcleos urbanos. Concretamente, para los centros educativos donde se requiere la navegación de varias máquinas simultáneamente, resulta una opción muy interesante y, ciertamente, superior al anterior sistema RDSI.

La combinación ADSL+Linux, como pretendemos ilustrar en este documento, presenta una potencialidad, impensable en otras plataformas. Linux es perfectamente compatible con la tecnología ADSL, es más, permite aprovechar el máximo de ancho de banda (2 Mb) ofertado actualmente por los distintos proveedores del servicio. Esta cuestión no es baladí. Curiosamente Microsoft Windows™ sólo puede aprovechar 1,5 Mb por la propia estructura del sistema operativo, aunque tuviesemos contratados 2 Mb.

De modo que la tecnología ADSL alcanza su pleno rendimiento utilizando Linux como plataforma. Otra cosa son los dispositivos de conexión. Por desgracia, algunos fabricantes, piensan antes en otros sistemas que en Linux,de momento, por lo que hay en el mercado ciertos dispositivos de conexión, sobre todo internos, que no están todavía soportados, a la espera que estas compañías faciliten la tarea de los desarrolladores de módulos. Sin embargo, la mayoría de dispositivos externos no presentan ningún problema para Linux, están perfectamente soportados y en disposición de navegar a través de ellos con el máximo de ancho de banda contratado.

7.2. La instalación del router

Al realizar la contratación de los servicios ADSL, el proveedor suele preguntarnos por el sistema operativo que vamos a usar, en caso de que no lo hiciesen, es el momento de dejar muy claro que la instalación será bajo Linux. No obstante, todos los dispositivos externos instalados habitualmente por Telefónica™ se encuentran perfectamente soportados.

Básicamente los routers de conexión ADSL tienen dos formas de operar, utilizando NAT y no ulilizándolo.[1] Telefónica™ ha bautizado con el nombre de *multipuesto* y *monopuesto* a las configuraciones usando NAT o no, respectivamente.

Cuando se usa "monopuesto", la IP pública está asociada al equipo y podemos acceder a todos los servicios de la red desde un sólo ordenador ya que no usamos NAT, o sea, es lo más parecido al uso del módem convencional de las conexiones RTB clásicas. Sin embargo, el "multipuesto" permite acceder con varias máqinas de nuestra intranet, asumiendo nuestro modem/router la IP pública y usando NAT para dirigir las peticiones.

En ocasiones, el técnico instalador nos preguntará sobre el tipo de configuración que deseamos para nuestro router. Pueder ser un buen momento para tomar una decisión y evitar así reconfigurarlo más tarde perdiendo la garantía del aparato.

En cualquier caso siempre es conveniente disponer de los datos más significativos de nuestra conexión ADSL, sobre todo la IP pública asignada a nuestra línea telefónica junto a su máscara de red, el resto de datos como DNS, DNS de gestión EDC, etc. suelen ser comunes a todos los usuarios. Si por alguna causa los hemos extraviado, siempre podremos solicitárselos a nuestro proveedor o tratar de averiguarnos nosotros mismos. Hay muchos sitios como http://www.adsl4ever.com que, además de ofertar información y software sobre la tecnología ADSL, nos permite realizar un test de velocidad sobre nuestra conexión, indicándonos nuestra IP pública, o sea, la usada para acceder a nuestro equipo desde el exterior. Sin embargo para obtener nuestra máscara de red, caso de funcionar con el router en multipuesto, sería bastante más complejo.

7.3. Nuestro router a medida

Las características de la tecnología ADSL combinadas con Linux, ya hemos adelantado, convierten a nuestro ordenador en una fabulosa herramienta. Pero para extraer el máximo rendimiento es necesario conocer con detalle todas las características del sistema, para adecuarlo a nuestras expectativas, aunque, de momento, ésto suponga ciertas contrariedades.

Cuando el concepto y la normativa sobre propiedad convencional se traslada al campo de software a causa de intereses meramente comerciales, surgen las contradicciones y las paradojas como consecuencia del afán recaudador de las insaciables compañías y resaltan la confusión y la inoperancia de un sistema legislativo, a todas luces, inadecuado para regular estas situaciones.

Actualmente hay diferentes proveedores de soluciones ADSL, pero en la práctica Telefónica™ es, de facto, el monopolio que siempre ha sido. Cuando un usuario contrata su conexión ADSL, adquiere el módem/router en propiedad, sin embargo le está prohibido acceder a su configuración, de modo que el proveedor instala el dispositivo ya configurado con los filtros que estima convenientes, sin tener en cuenta al usuario que ha adquirido el aparato.

Normalmente se suelen instalar con dos "filtros" para bloquear los accesos a nuestro router, uno desde Internet y otro desde nuestra red local. Pero, aún resulta más controvertido si queremos acceder al router, de nuestra propiedad, a través de un puerto serie puesto que necesitamos un "usuario" y una "clave" para poder interactuar con él. Sin embargo estos datos no son facilitados por el proveedor. Esta situación carece totalmente de sentido y lo lógico sería que el propietario del router conociese a fondo todo lo relativo a la configuración de su dispositivo ya que, al fin y al cabo, se trata de su propia seguridad. En cambio nos encontramos que el usuario es tratado como un completo ignorante y sujeto a unas condiciones unilaterales que le suponen, actualmente, la pérdida de garantía en el momento que decida configurarse su módem/router.

7.3.1. Cambio de multipuesto a monopuesto

Retocar la configuración que, por defecto, posea nuestro módem/router es necesario para aprovechar al máximo las opciones que nos brinda la banda ancha. Dependiendo del modelo de dispositivo que tengamos instalado puede cambiar algo la forma pero no el fondo de la personalización necesaria. Vamos a ilustrar el proceso para un router externo Speed Efficent 5660, uno de los habituales instalados por Telefónica™ si usted dispone de otro modelo, los pasos a seguir, serán probablemente similares, de todas formas una búsqueda suficientemente "fina" en Google (http://www.google.com) le proporcionará una valiosa colección de enlaces a documentación específica de su dispositivo concreto.

> **Importante:** Recuerde, antes de hacer ningún cambio en la configuración de módem/router, que puede suponer la pérdida de la garantía del aparato.

Lamentablemente los fabricantes y los proveedores suelen utilizar, de momento, software de gestión basado en plataformas propietarias como Microsoft Windows™, sin embargo, los usuarios de Linux también tenemos soluciones libres alternativas. Concretamente usaremos dos, accediendo a través del puerto serie y mediante el pequeño servidor web que lleva incorporado el dispositivo.

7.3.1.1. Acceso a través del puerto serie

La herramienta GNU (http://gnu.org) que nos permite el acceso a través del puerto serie es microcom, que no se instala por defecto, así que debemos utilizar los discos de instalación de nuestro sistema, o bien, obtenerla de su sitio

web (http://microcomLinux.homestead.com/files/microcom.html) donde encontraremos, además de la aplicación y sus fuentes, documentación y ejemplos de uso.

Para conectarnos usaremos

```
./microcom -D /dev/ttyS0
```

Si tenemos accesible el router a través del primer puerto serie.[2]

Probablemente nos pregunte por un usuario y una clave para el acceso. Habitualmente Telefónica™ usa **adminttd** para ambos campos. En cualquier caso, esta información circula abundantemente por Internet. Una vez dentro nos facilitará una interfaz de comandos para realizar la tarea de configuración en modo texto. Pulsaremos varias veces la tecla **Enter** hasta que se nos solicite una contraseña. En ese instante, desconectamos el router para que se resetee y al reiniciar debemos teclear **default**, entonces debe aparecer en pantalla:

```
Command->
```

Lo que nos indica que ya nos encontramos interactuando con el router.

Ahora debemos restaurar la configuración original y establecer una nueva clave de acceso, para ello tecleamos:

```
Command->default sll
Command->set password
Command->reboot
```

La configuración de fábrica suele venir sin contraseña, pero en caso de que se nos solicitase debemos probar con "adminttd", "default" o "root". Tras reiniciar debemos contestar "si" a la pregunta sobre seguridad que se nos formula e introducir la nueva clave de acceso que hemos fijado y comenzaremos a configurar el sistema a nuestra medida.

En primer lugar, fijaremos el nombre para el dispositivo[3] y desactivaremos el bridge y el DNS para usar el de Telefónica™ de la siguiente forma:

```
Command->set hostname 950xxxxxx
Command->set bridge disable
Command->set dns disable
DNS changes will take place after reboot
```

Bien, vamos a fijar el modo "monopuesto" para el router. Supongamos que la IP pública de nuestra ADSL es 20.20.20.20 con máscara de red 255.255.255.128, calculamos la nueva IP pública (IP pública AND máscara)+1, resultando como nueva IP pública del router 20.20.20.129, establecemos el tipo de conexión (RFC1483) y la dirección privada del router a efectos de acceso desde la red interna, desactivamos RIP y NAT puesto que son propios del modo "multipuesto" y finalmente reiniciamos el router. La secuencia de comandos sería:

```
Command->set ethip
20.20.20.129 255.255.255.128
Implement IP changes now?
default: n [y,n] y
Command->set vc 1483r 8 32 llc
max 192.168.1.7 255.255.255.128
Changing the VC type requires a reboot
Command->set ipgateway 192.168.1.0
Warning: Saved IP gateway address is currently unreachable
command ipgateway: failed
Command->set ripcfg none
```

```
Rip change will take place after reboot
Command->set napt disable
Command->reboot
Are you sure?
default: n [y,n] y
```

Ya sólo resta configurar nuestro servidor **gato** con el gateway de la tarjeta de red `eth1` apuntando a la IP pública del router 20.20.20.129 para que podamos navegar desde nuestra red y confiando la seguridad de la misma a nuestro firewall instalado en el servidor. Ahora será responsabilidad verdaderamente nuestra.

7.3.1.2. A través de administración web

Este router lleva incorporado un pequeño servidor provisto de una herramienta de configuración web. Para acceder a ella basta teclear en la ventana del navegador la dirección IP de gestión, introducir el usuario y la contraseña antes descritos para proceder a realizar los ajustes necesarios y dejar nuestro router trabajando en "monopuesto" con las características señaladas en el apartado anterior.

Al tratarse de una herramienta de gestión web eminentemente grafica e intuitiva, consideramos que el proceso no precisa demasiada minuciosidad, además, hay una magnífica documentación en castellano que describe con detalle todas las opciones de configuración para este router. Veamos, pues un repaso somero por los puntos más importantes de la configuración.

- En las secciones `NAPT Mode` e `IP Filter Mode` hemos de seleccionar `Disable` y posteriormente `Change Mode`, esto desactivará los filtros y el protocolo NAT.
- En la sección `DSL WAN IP Address` residen los datos de conexión. Es conveniente anotarlos ya que son la IP pública y la máscara de red de nuestro router. Luego, adjudicaremos los valores `VPI=8 VCI=32 1484Routed LLC Max Rate` accediendo en el botón **1483R**. El campo IP Address lo cumplimentaremos con nuestra propia IP de gestión del módem para acceder a su configuración desde la intranet, mientras que la máscara de red la cumplimentaremos con la misma que posee la IP pública validando con **Modify VC**.
- Volviendo al menú anterior fijamos como `Default IP Gateway` el valor `192.168.1.0` y validamos con `Set New Value`
- Finalmente regresamos y rellenamos el campo `Ethernet LAN IP Address` con la nueva IP obtenida según la sección anterior, es decir, `20.20.20.129`, manteniendo la misma máscara de red, desactivamos la casilla de verificación `Do not let change effect until next reboot` para que el cambio se aplique inmediatamente y confirmamos con`Set New Values` perdiendo así contacto con el router que pasa a tener como IP de gestión la que introdujéramos en el segundo punto del proceso.

De esta forma ya tenemos nuestro sistema funcionando bajo nuestro control. Es el momento de sacarle partido.

Notas

1. Véase la Sección 3.2
2. Recordemos que en Linux los puertos serie se identifican como `/dev/ttyS0, /dev/ttyS1,...` lo que en Windows™ se conoce como COM1, COM2,... y así sucesivamente.
3. Telefónica™ suele usar como nombre nuestro número telefónico, pero podemos bautizarlo como nos apetezca.

Capítulo 8. Rentabilizando la configuración

8.1. Precauciones de seguridad

Todo el proceso descrito para el montaje de nuestro servidor **gato** sólo tendría sentido académico, experimental o mera curiosidad intelectual, si la tarea finalizase aquí. Sin embargo, el objetivo final persigue la rentabilidad de la instalación en un centro educativo, aprovechando las enormes posibilidades que nos brindan las soluciones de código abierto para beneficio de la comunidad escolar.

Procede, pues, utilizar nuestro servidor para que cumpla su función alojando y proporcionando material, tanto para Internet, como para nuestra propia intranet. Ello precisa que los miembros de la comunidad tengan un tipo de acceso especial al servidor que no sea el usuario pasivo que conecta con él y consulta los documentos web, sino que deben tener la posibilidad de alojar sus propios contenidos aprovechando así las capacidades del sistema instalado.

Este tipo de acceso, vamos a llamarle "privilegiado", tiene ciertos riesgos que no debemos soslayar si pretendemos que nuestro servidor, y por ende nuestra red, cuente con ciertas garantías de inviolabilidad.

Cualquiera que posea una cuenta de acceso a una máquina Linux suele contar con una serie de privilegios susceptibles de constituir un peligro potencial para la seguridad del sistema. Tal vez el que entraña un mayor riesgo sea la "shell de comandos" ya que permite, lógicamente, al usuario interactuar con la máquina y ejecutar aplicaciones en ella. Igualmente, el espacio en disco puede ser saturado por un usuario malicioso y, si no tomamos las medidas adecuadas, causar un trastorno grave en el mantenimiento del sistema.

Por otra parte, cada puerto que tengamos abierto es un peligro potencial que puede comprometer la seguridad, según dicen en letras bien grandes todos los manuales de administración, sin embargo estamos obligados a mantener algunos abiertos como el 80 (http) o el 21 (ftp) si queremos, efectivamente, "servir web". Por eso hemos de ser muy cautelosos, mantener accesos sólo en los puertos imprescindible y estar muy atentos a las actualizaciones de seguridad de los "demonios" que filtran dichos puertos, procurando utilizar las últimas versiones de los mismos.

8.2. Creando los usuarios

Al crear un usuario de un sistema Linux el comando **adduser** pone a disposición de éste, por defecto, un directorio que normalmente se encuentra en `/home/usuario` junto con una shell de comandos que le permiten ejecutar aplicaciones en la máquina. Lógicamente, todos los usuarios normales de **gato** podrían acceder a él y usarlo para alojar sus documentos web.

No parece ser ésta una solución adecuada a nuestra situación. Si creamos una cuenta para cada miembro de la comunidad, éste podría publicar en el servidor, pero también podría hacer muchas más cosas. Algunas no deseadas. Con el agravante de que, con el transcurso del tiempo, nuestro servidor tendría multitud de usuarios dispersos por toda la geografía, española cuando menos, con sus contraseñas fuera de nuestro control y a nadie se le escapa el peligro potencial de esta situación.

Nos planteamos entonces cómo permitir el acceso con posibilidad de alojar documentación e impedir todo lo demás a la gran mayoría de nuestros usuarios. Bien, la solución definitiva la desconocemos, sin embargo hemos usado una estrategia que funciona, aunque sin duda, habrá otras más elegantes y refinadas para conseguir este propósito.

8.2.1. Cuotas de disco

En primer lugar debemos limitar el espacio en disco para los usuarios de nuestro sistema.

8.2.2. Directorios personales

Ahora debemos proporcionar a cada usuario un espacio en nuestro disco y la posibilidad de escribir en él para que así pueda publicar sus trabajos en la web.

Creamos un grupo llamado **huesped** al que pertenecerían todos aquellos que tuviesen alojamiento web.

Los miembros de este grupo no podían tener una shell válida y, además, su espacio en disco debería estar situado bajo el control de Apache, puesto que la única finalidad era el hospedaje de sus documentos html.

La tarea de crear usuarios en estas condiciones decidimos automatizarla creando un simple *script* llamado `adhu` que reproducimos a continuación:

```
# Este script pretende automatizar la entrada
# de usuarios que tienen alojamiento web
# Se añaden al grupo huesped que tiene que estar creado
# tiene que crearse tambien un esqueleto /etc/skel2
# optativamente puede contener dos ficheros .welcome.msg y .quit.msg
# pedimos el nombre de usuario:
#! /bin/bash
echo -n "Intoduzca el nombre de usuario: "
read HUESPED
#Comprobamos si el usuario existe
if grep "^$HUESPED" /etc/passwd >/dev/null 2>&1

then
     echo "EL USUARIO YA EXISTE"
else
echo "CREANDO EL USUARIO $HUESPED"
#Añadimos el usuario
adduser -d /var/www/html/$HUESPED -m -k /etc/skel2 -g huesped $HUESPED
echo "Cambiando los permisos del directorio"
chmod 755 /var/www/html/$HUESPED
passwd $HUESPED
echo "USUARIO CREADO"
fi
```

Este script está suficientemente comentado, sin embargo vamos a insistir en la línea más relevante, tal vez.

`adduser -d /var/www/html/$HUESPED -m -k /etc/skel2 -g huesped $HUESPED`

El comando **`adduser`** empleado con los modificadores descritos, crea el directorio del usuario bajo `/var/www/html`[1], asigna automáticamente al usuario como miembro del grupo **huesped** y, tal vez lo más importante, la expresión **`-m -k /etc/skel2`** hace posible que el usuario no disponga de una shell de comandos, puesto que el esqueleto definido en `/etc/skel2` está vacío, bueno, en nuestro caso tiene los mensajes de bienvenida y despedida en sendos archivos ocultos. Este es su contenido:

```
drwxr-xr-x    2 root     root         4096 jun  3  2002 .
drwxr-xr-x   55 root     root         8192 mar  1 18:26 ..
-rw-r--r--    1 root     root          188 jun  3  2002 .quit.msg
-rw-r--r--    1 root     root          133 jun  3  2002 .welcome.msg
```

De no ser así, por defecto leería el contenido de la plantilla albergada en el esqueleto ubicado en `/etc/skel` que contiene normalmente lo siguiente:

```
drwxr-xr-x    3 root     root         4096 jun  3  2002 .
drwxr-xr-x   55 root     root         8192 mar  1 18:26 ..
-rw-r--r--    1 root     root           24 jul  9  2001 .bash_logout
-rw-r--r--    1 root     root          191 jul  9  2001 .bash_profile
-rw-r--r--    1 root     root          124 jul  9  2001 .bashrc
-rw-r--r--    1 root     root          118 ago 10  2001 .gtkrc
drwxr-xr-x    3 root     root         4096 jun  3  2002 .kde
-rw-r--r--    1 root     root         3511 ago  3  2001 .screenrc
```

Notas

1. Esto es opcional, ya que también podríamos indicarle al servidor Apache otro directorio raiz para colocar las webs personales. Sin embargo es una buena costumbre respetar los estándares e incluso situar el directorio `/var` como una partición independiente, lo que facilita las copias de seguridad y reduce el peligro en caso de catástrofe.