Universidad Austral de Chile
Facultad de Ciencias de la Ingeniería
Escuela de Electricidad y Electrónica

Generación de un laboratorio de redes de computadores bajo Linux para la Compañía Telefónica del Sur

Trabajo de Titulación para optar al Título de Ingeniero Electrónico.

Profesor Patrocinante:
Sr. Pedro Rey Clericus
Ingeniero Electrónico

FRANCISCO JAVIER MORAGA QUEZADA
VALDIVIA 2003

COMISION EXAMINADORA

PROFESOR PATROCINANTE

Sr. Pedro Rey C.

PROFESOR INFORMANTE

Sr. Néstor Fierro M.

INGENIERO INFORMANTE

Sr. Edgardo Herrera P.

AGRADECIMIENTOS

A Dios por la vida y todo lo que ella me ha entregado. En cuanto a las personas a quienes debo el logro y el conocimiento que he alcanzado en este tiempo, mis padres Francisca y Jubel, mi hermana Pamela y cada persona que estuvo con un gesto o palabra de aliento y cariño en los momentos que lo necesitaba; muchas gracias de todo corazón.

Gracias profesores por el respeto y la entrega que siempre estuvo presente en las enseñanzas que recibí de parte de todos ustedes.

Claudia, que la vida nos entregue lo mejor y Dios nos acompañe en el caminar que nos hemos propuesto juntos, gracias por cada momento de felicidad y amor que he recibido

También deseo agradecer muy especialmente a mi tutor por parte de Telefónica del Sur, Ing. Edgardo Herrera P, por su valioso aporte, apoyo y dirección en la concreción de este trabajo.

INDICE

CAPITULO II: INSTALACION

CAPITULO III: TRABAJANDO CON LINUX

RESUMEN

En el presente trabajo se realiza una revisión de los principales pasos para instalar y configurar un equipo con sistema operativo Linux, aplicado a los servicios de conectividad tales como, Web, Ftp, DNS entre otros.

El objetivo es que cualquier Ingeniero o Técnico, pueda acompañado de este documento realizar un análisis de cómo abordar la configuración de un servidor Linux y tener una herramienta para analizar problemas de los servicios configurados a los clientes relacionados con este sistema operativo.

La metodología que se aborda en el trabajo consiste en recopilar un conjunto de pasos relevantes de la configuración y administración de un servidor Linux, formando grupos temáticos, lográndose una cobertura conceptual integrada y completa acompañada con una serie de laboratorios, los cuales se detallan dentro de los Objetivos específicos.

A partir del capítulo 6 se comienza a tratar el tema de Interconectividad de una manera más específica, presentada como Redes Bajo Linux, Enrutamiento, Cortafuegos y Servidores Proxy.

Para esta presentación se utilizó la distribución RedHat, aún cuando es posible de aplicar bajo cualquier otro tipo de distribución.

SUMMARY

On the following work, it has been done a revision of the main steps to install and configure equipment with Linux as its Operative System, focus on connectivity services such as Web, Ftp, and DNS.

The objective is that any Engineer or Technician can with this document make an analysis of how to aboard the configuration of a Linux Server and to have a tool to figure up problems with the servers' on clients.

The methodology on this work consist of getting the most relevent steps of the configuration and administration of a Linux Server, forming thematic groups, reaching a complete and integral conceptual approach followed with a series of laboratories, who are detailed on the Specific Objective part.

From chapter six it begins to cover the interconectivity theme on a very specific way, presented as Linux Networks, Routing, Firewalls and Proxy Servers.

For this work it was used the Red Hat distribution, any way it is possible to apply on any type of distribution.

INTRODUCCION

Linux es un clon de Unix libremente distribuible para computadores personales. Actualmente corre en una amplia variedad de máquinas, que incluye no sólo a la familia Intel, sino también las máquinas basadas en arquitecturas Motorola 680x0, como el Commodore Amiga o el Apple Macintosh. También están soportadas las máquinas Sun SPARC y Ultra-SPARC, las Compaq Alpha, MIPS, PowerPCs (Mac de última generación) e incluso StrongARM (dispositivos de mano o *handhelds*). Además, existe reescritura de Linux en plataformas más "complejas" como el IBM S/390 o los Fujitso AP-1000. Y tal parece que esta tendencia no va a decaer.

Linux fue desarrollado por un gran equipo de voluntarios a través de Internet. Fue iniciado en 1990 por el estudiante finlandés Linus Torvalds, como un proyecto universitario. Desde entonces, ha ido creciendo hacia un sistema compatible-unix muy completo, y muy utilizado para todo tipo de aplicaciones, desde simples procesadores de texto hasta avanzados sistemas de reconocimiento de voz, sin contar con que es la plataforma perfecta para acceder a Internet. Además soporta una gran cantidad de hardware, y tiene una pila TCP/IP completa, con adición de software cortafuegos, implementaciones PPP, SLIP, IPX; e incluso protocolos que otros sistemas no suelen incluir. Linux es potente, rápido, libre, y su popularidad sigue aumentando.

El sistema operativo Linux está cubierto por la Licencia Pública General GNU, que es la misma que rige el software desarrollado por la Free Software Foundation. Esta licencia permite a cualquiera modificar y/o redistribuir el software (gratuitamente o a cambio de dinero), «contagiándose» de dicha licencia cualquier modificación posterior. El término inglés «free» se refiere a la libertad, no a su precio, o sea no necesariamente gratis.

Como hoy en día, la mayoría de las compañías proveedoras de soluciones RED busca abaratar los costos de sus insumos, sin descuidar la QoS, Linux se presenta como una alternativa sólida para el manejo de redes de datos.

Desde hace varios años se ha estado utilizando esta plataforma para la supervisión de redes, por lo cual no es de extrañar que varias compañías lo vean como un sistema operativo de excelencia para este tipo de entorno. El mito que se tiene de Linux es su complejidad a la

hora de instalarlo y administrarlo, por lo que muchas veces se le critica, pero esta "complejidad" (que cada vez va desapareciendo) puede ser su característica más valiosa ya que una vez dominado, permite al administrador un control total en los procesos que se van llevando a cabo.

Objetivos Generales

El siguiente trabajo buscará poner a flote los beneficios de Linux para la re-ingeniería. La idea es que teniendo computadores obsoletos (386 o 486) se puedan "reciclar" y buscarles nuevos usos, como estaciones delegadas o como estaciones de tránsito de redes privadas.

Si tiene en cuenta que Linux es de libre distribución, según las bases de la GNU, es obvio el significado económico que implica una red de estas características para una compañía que quiera darle una solución para una PYME.

Con todo lo anterior, se puede configurar una red o sub-red ocupando un computador 386-486 como servidor propiamente tal, levantando los servicios de servidor de web, correo, FTP, DNS.

Objetivos Específicos

Como objetivo específico se buscará a través de este trabajo otorgar a la Compañía Telefónica del Sur una Guía práctica de Linux para los profesionales relacionados con el tema, en conjunto con una serie de laboratorios, divididos de la siguiente manera:

Lab 1:

a) Hardware Soportado antes de Instalar
b) Tipos de Instalaciones
c) Expansión del Kernel Paso a Paso
d) Entornos de Usuario

Lab 2:

a) Fontanería en Linux

b) Permisos y enlaces de Ficheros
c) Comandos de Administración y Gestión

Lab 3:
a) Configuración de la Red
b) Demonio Correo
c) Demonio FTP
d) Demonio Web

Lab 4:
a) Cortafuego con IPTables
b) Servidor Proxy
c) Herramientas de Linux; TCPdump, MRTG.

Temas a Tratar

El orden en el cual se plantearán los temas relacionados a este trabajo, se inician con los primeros capítulos 100% teóricos para luego, pasar a experiencias de laboratorio definidas anteriormente.

Primer Capítulo
1) Breve historia de Linux
2) Situación actual de Linux en el mundo
3) Linux, Windows NT y sus diferencias

Segundo Capítulo
1) Previo a instalar Linux
2) Particiones
3) Creación de de los Sistemas de Ficheros
4) Instalación

Tercer Capítulo

1) Entorno de usuarios
2) Ficheros y directorios
3) Primeros pasos
4) Sumario de órdenes

Cuarto Capítulo

1) Sistemas de ficheros
2) Interpretes de comandos
3) Fontanería, permiso y manejo de ficheros
4) Tareas y procesos

Quinto Capítulo

1) Servicio Web (Apache)
2) Servicio DNS (BIND 8)
3) Correo Electrónico
4) FTP

Capítulo Seis

1) TCP/IP
2) Cortafuegos
3) Servidores Proxy
4) Routing avanzado
5) Utilidades para la administración de Redes (Tcpdump, MRTG)

Estos Laboratorios estarán enfocados a la parte de Interconectividad de una red experimental bajo Linux, y la medición de su desempeño se puede efectuar con la ayuda de Tcpdump, donde los resultados finales determinaran las ventajas comparativas de este modelo.

CAPITULO I
ANTECEDENTES GENERALES

1. Breve historia de Linux

UNIX es uno de los sistemas operativos más populares del mundo debido a su extenso soporte y distribución. Originalmente fue desarrollado como sistema multitarea con tiempo compartido para mini computadores y mainframes a mediados de los 70', y desde entonces se ha convertido en uno de los sistemas más utilizados a pesar de su ocasionalmente, confusa interfaz con el usuario y el problema de su estandarización.

Ahora, ¿cual es la verdadera razón de la popularidad de UNIX? Muchos hackers consideran que UNIX es el autentico y único sistema operativo. El desarrollo de Linux parte de un grupo en expansión de hackers de UNIX que quisieron hacer su sistema con sus propias manos.

Existen numerosas versiones de UNIX para muchos sistemas, desde computadores personales hasta supercomputadores como el Cray Y-MP, la mayoría de las versiones de UNIX para computadores personales son muy caras, una copia para una máquina 386 del UNIX System V de AT&T cuesta unos 1500 dólares estadounidenses.

Linux es una versión de UNIX de libre distribución, inicialmente desarrollada por Linus Torvalds[1] en la Universidad de Helsinki, en Finlandia. Fué desarrollado con la ayuda de muchos programadores y expertos de UNIX a lo largo y ancho del mundo, gracias a la presencia de Internet. Cualquier habitante del planeta puede acceder a Linux y desarrollar nuevos módulos o cambiarlos a su antojo. El núcleo de Linux no utiliza ni una sola línea del código de AT&T o de cualquier otra fuente de propiedad comercial, y buena parte del software para Linux se desarrolla bajo las reglas del proyecto de GNU de la Free Software Foundation, Cambridge, Massachusetts.

Inicialmente, sólo fue un proyecto de aficionado de Linus Torvalds. Se inspiraba en Minix, un pequeño UNIX desarrollado por Andy Tanenbaum, y las primeras discusiones sobre Linux surgieron en el grupo de News comp.os.minix. Estas discusiones giraban en torno al desarrollo de un pequeño sistema UNIX de carácter académico dirigido a aquellos usuarios de Minix que querían algo más.

[1] torvalds@kruuna.helsinki.fi.

El desarrollo inicial de Linux ya aprovechaba las características de conmutación de tareas en modo protegido del 386, y se escribió todo en ensamblador, Linus dice:
"Comencé a utilizar el C tras escribir algunos drivers, y ciertamente se aceleró el desarrollo. En este punto sentí que mi idea de hacer un “un Minix mejor que Minix” se hacia más seria. Esperaba que algún día pudiese recompilar el gcc bajo Linux. . .
"Dos meses de trabajo, hasta que tuve un driver de discos (con numerosos bugs pero, que parecía funcionar en mi PC) y un pequeño sistema de ficheros. Aquí tenía ya la versión 0.01 [al final de Agosto de 1991]: no era muy agradable de usar sin el driver de disquetes, y no hacía gran cosa. No pensé que alguien compilaría esa versión."

No se anunció nada sobre esa versión, puesto que las fuentes del 0.01 jamás fueron ejecutables, contenían sólo rudimentos de lo que sería el núcleo, y se asumía que se tenía que tener acceso a un Minix para poderlo compilar.

El 5 de Octubre de 1991, Linus anunció la primera versión "oficial" de Linux, la 0.02. Ya podía ejecutar bash (el shell de GNU) y gcc (el compilador de C de GNU), pero, aún estaba en sus inicios. La intención era ser una herramienta para hackers. No había nada sobre soporte a usuarios, distribuciones, documentación. Hoy, la comunidad de Linux aún trata estos asuntos de forma secundaria. Lo primero sigue siendo el desarrollo del kernel.

Linus escribía en comp.os.minix[2]

"Suspiras al recordar aquellos días de Minix-1.1, cuando los hombres eran hombres y escribían sus propios drivers? Te sientes sin ningún proyecto interesante y te gustaría tener un verdadero S.O. que pudierais modificar a placer? Te resulta frustraste el tener solo a Minix? Entonces, este artículo es para ustedes….
Como dije hace un mes, estoy trabajando en una versión gratuita de algo parecido a Minix para computadores At-386. He alcanzado la etapa en la que puede ser utilizable y voy a poner las fuentes para su distribución. Es solo la versión 0.02… pero he conseguido ejecutar en el: bash, gcc, gnu-make, gnu-sed, compress, etc.”

Tras la versión 0.03, Linus saltó a la versión 0.10, al tiempo que más gente empezaba a participar en su desarrollo. Tras numerosas revisiones, se alcanzó la versión 0.95, reflejando la

[2] Grupo de noticia que aún se discuten temas relacionados con Minix

esperanza de tener lista muy pronto una versión "oficial" (Teóricamente, la versión 1.0 de los programas corresponde con la primera completa y sin errores). Esto sucedía en Marzo de 1992. Año y medio después, en Diciembre del 93, el núcleo estaba en la revisión 0.99.pl14, en una aproximación al 1.0.

Hoy Linux es ya un clónico de UNIX completo, capaz de ejecutar X Window, TCP/IP, Software de correo, FTP, WEB, etc. Mucho software de libre distribución ha sido ya portado a Linux, y están empezando a aparecer aplicaciones comerciales. El hardware soportado es mucho mayor que en las primeras versiones del núcleo. Mucha gente ha ejecutado tests de rendimiento en sus sistemas Linux 486 y se han encontrado que son comparables a las estaciones de trabajo de gama media de Sun Microsystems y Digital. ¿Quien imaginaría que este "pequeño" clónico de UNIX iba a convertirse en un estándar mundial para los computadores personales?

2. Situación Actual de LINUX en el Mundo

En la actualidad la mayor parte de los usuarios de un computador utilizan el sistema operativo Windows en cualquiera de sus versiones como base, esto sucede en realidad porque la mayoría de las máquinas que compra, vienen precargadas con este tipo de software, sin embargo, dependiendo del área en el que se desenvuelva el profesional serán las necesidades de seguridad, desempeño y herramientas que debe tener. Por ejemplo, para un diseñador gráfico será mejor utilizar un ambiente basado en *mac* y para una secretaria que solamente utiliza la máquina como procesador de texto puede tener única y exclusivamente un sistema basado en *windows9x o windows2000,* sin embargo, para un desarrollador de software o alguna empresa que busque seguridad en su red será mejor tener alguna distribución de *Unix*, como puede ser *Linux, RedHat, Suse, Mandrake*, etc., hace algunos años eran muy pocos los usuarios de otras plataformas distintas a Windows y, aunque la mayor parte todavía utiliza como *default* Windows, cada vez son más los usuarios y empresas que migran a otras plataformas, he ahí la importancia de capacitar a los profesionales que trabajan en el área de la telecomunicaciones y que interactúan con clientes.

2.1 Usuarios de linux en el mundo

Las tres plataformas principales como ya se ha mencionado son Windows, Linux y Mac, en cuanto a sistemas operativos personales y a sistemas operativos de red Windows NT, Linux y NetWare.

En Chile, las empresas ligadas al mundo de las telecomunicaciones están reconociendo a Linux y ya se están creando grupos importantes de trabajo, aunque el mayor porcentaje de éstos siguen con Windows. Las empresas más importantes, así como el gobierno están cambiando de plataforma.

Los sistemas operativos se definen como el conjunto de procedimientos que permiten el control de los recursos de una instalación para el procesamiento de datos. Los recursos comprenden el equipo, los programas, los datos y el o los operadores. Sin embargo, por lo general se define un sistema operativo como a *un programa que supervisa la operación de otros programas*. Estos programas son de tres tipos básicos: control de tareas, sistema de control de entrada-salida y programa de procesamiento.

Una vez definido lo que es un sistema operativo se puede mencionar que hay diversos sistemas o plataformas, los más conocidos son MAC, UNIX (en diversas versiones) y Windows. Cada uno tiene diversas características, las cuales serán presentadas a continuación:

UNIX: Sistema desarrollado desde hace 35 años, manteniendo el mismo diseño y forma de uso, añadiendo diversas mejoras a lo largo de los años, pero manteniéndose sobre la línea de seguridad y alto rendimiento sobre la cual fue diseñado.

TCP/IP: Protocolo básico de Internet, fue construido alrededor de UNIX, por lo tanto, la integración de los servicios de Internet en un ambiente UNIX es perfecta.

LINUX: Sistema operativo libre, similar a UNIX, desarrollado colectivamente por miles de programadores en todo el mundo desde 1991, evolucionando de un proyecto de programación de un par de personas a un sistema empleado por (estimado) 10 millones de personas.

WINDOWS NT: Sistema operativo desarrollado a partir de 1992, para brindar seguridad a redes basadas en sistemas personales Windows, por lo cual requiere mantener compatibilidad y

coherencia de interfaz con éstos. Se desarrolló alrededor del esquema de red SMB (Windows para grupos/red Microsoft) y posteriormente, se le agregó soporte para TCP/IP.

MAC OS/X: El sistema operativo Darwin[3] ofrece una configuración protegida de la memoria que afecta un espacio único para cada aplicación. Cuando las aplicaciones se aíslan en su propia memoria, no se necesita recomenzar el computador en caso de error, Darwin cancela la aplicación, dejándole continuar trabajando o seguir funcionando sin la interrupción.

3. LINUX, WINDOWS NT y sus diferencias

3.1 ¿Por qué comparar dos sistemas operativos tan diferentes?

Ambos apuntan al mismo mercado, que es el de más rápido crecimiento a nivel empresarial: Servidores de red y estaciones de trabajo de alta capacidad, que es hacia donde está dirigido este trabajo.

Las capacidades de las computadoras personales y de las Workstations van siendo cada vez más similares.

Microsoft, la empresa que desarrolló Windows NT, se ha convertido en un monopolio, frente al cual únicamente los sistemas Unix (y derivados) son alternativas viables.

[3] Corresponde a una versión de Sistema operativo MAC

3.2 Paralelo entre Linux y Windows NT

3.2.1 Seguridad

Tabla 1: Comparación de Seguridad entre Linux y Windows

LINUX	Windows NT
Seguridad en los archivos: Cada archivo tiene definida la seguridad para Dueño, Grupo y Otros. Cada uno de ellos tiene permisos de lectura, escritura y ejecución. Sabiendo manejar este sistema, presenta una gran flexibilidad. Además de esto, tiene la característica del *SUID bit*, que permite que un archivo se ejecute con la identidad de un usuario determinado diferente del usuario que lo ejecuta.	Seguridad en los archivos: A cada archivo se puede asignar varios grupos de atributos basados en usuarios o grupos, con permisos de creación, lectura, escritura, remoción, y ejecución, creando fácilmente listas de control de acceso (*ACLs*).
Maneja todos los niveles del sistema, un verdadero sistema de multiusuario, permitiendo nativamente que se puedan conectar simultáneamente diferentes usuarios, y manteniendo los recursos que ocupan cada uno de ellos perfectamente aislados de los procesos de otros usuarios.	No existe realmente el concepto de Multiusuario, aunque ha habido intentos de lograrlo, nunca han sido exitosos; Esto es, en buena parte, por una deficiente administración de memoria y recursos, y por utilizar un diseño de sistema operativo como servidor de archivos únicamente, no como servidor de aplicaciones.
TCP/IP fue desarrollado sobre UNIX, por tanto su implementación es la más segura y ampliamente probada. Si una operación ilegal llega a bloquear el subsistema de TCP, este típicamente se reestablece tan pronto esta operación termina.	El subsistema de TCP/IP para Windows fue creado para seguir, en la medida de lo posible, los estándares; sin embargo, en caso de haber operaciones ilegales, es muy raro que el sistema pueda continuar operando, pues casi siempre el resultado es que Windows cae en la "pantalla azul de la muerte", situación tras la cual hay que reiniciar forzosamente el sistema a mano.
Al ser Linux software libre, no pasan normalmente más de un par de horas entre	Windows está basado en el esquema de "seguridad a través de la oscuridad": El usuario

que es encontrado un error y que éste es corregido y la corrección publicada. Esto hace que el impacto de cualquier problema de seguridad sea mínimo. Desde 1996, cuando fue liberado el Kernel (núcleo) 2.0.0, se liberaron 38 revisiones a este. El Kernel 2.2.0 fue liberado a principios de marzo de 1999, y (a fines de 1999) ya va en su decimosegunda revisión.	no tiene acceso al código, por tanto no le es tan fácil encontrar errores. Sin embargo, cuando estos llegan a ser encontrados no aparece un parche sino hasta meses después, con el "service pack" correspondiente. Desde 1996, fecha en que apareció Windows NT 4.0, sólo han sido publicados 6 *service packs* - el último de ellos pesa mas de 30MB. Un caso muy notorio de los problemas que puede causar la seguridad a través de la oscuridad es el que se dio a conocer el 14 de abril del 2000, tras más de dos años de existencia - En todos los servidores Windows NT con extensiones de Frontpage 98 viene una puerta trasera secreta, con la contraseña *Netscape enigeers are weenies*, afectando a millones de servidores en todo el mundo.

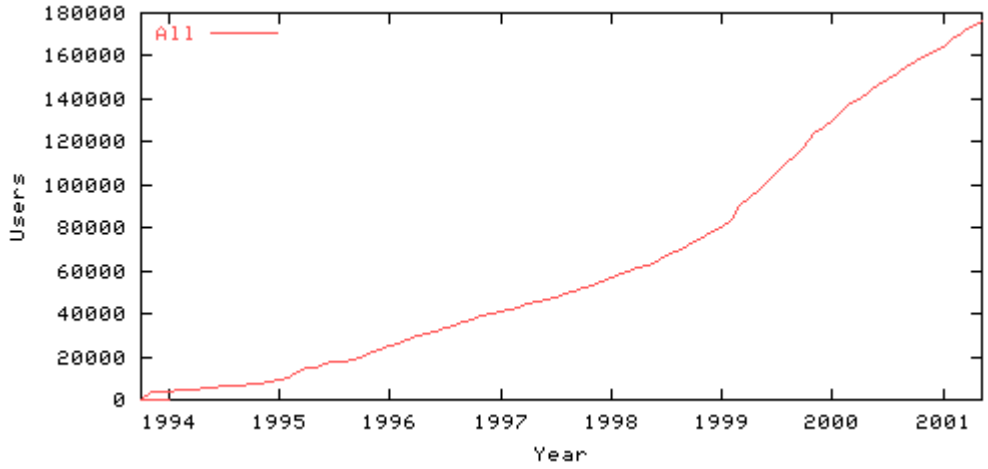
3.2.2 Facilidad de Uso

Tabla 2: Comparación de la Facilidad de Uso entre Linux Windows NT

LINUX	Windows NT
Curva de aprendizaje un tanto más pronunciada, aunque esta tiende a suavizarse gracias a proyectos como Linuxconf, GNOME, KDE o GNUstep, que son programas de configuración gráfica, y no de comandos de terminal, lo que crea un ambiente más amigable, similar a Windows.	Saber utilizar Windows es todo lo necesario para utilizar como usuario a Windows NT; sin embargo, para saber administrar un sistema NT hace falta conocer mucho más, ya que las herramientas no siempre están donde se espera, ni reaccionan como se supone. Poniéndolo en otras palabras, es verdad lo que dice Microsoft: Windows NT es *un sistema que cualquier niño puede utilizar*. Ahora bien, ¿Quiere que cualquier niño administre el sistemas a su antojo?

Filosofía UNIX para las diversas partes del sistema: Una gran cantidad de pequeñas herramientas muy especializadas y fáciles de integrar en scripts, automatizando operaciones completas fácilmente.	Filosofía Windows para las diversas partes del sistema: Pocas herramientas grandes y poderosas, hechas para resolver situaciones definidas y completas, y muy difíciles de integrar para lograr soluciones a problemas específicos.
Muchas empresas no cuentan con únicamente un servidor. Muchas veces, los cuartos de servidores son lugares aislados del personal en general, llenos de costoso equipo. Otras veces, la compañía tiene servidores dedicados a diferentes sucursales. Por estas y otras muchas causas, es comúnmente muy deseable que el o los administradores puedan realizar sus labores remotamente. En sistemas Linux/UNIX, este punto no tiene que planearse siquiera – Todos los programas del sistema correrán sin ninguna diferencia de manera local o remota, gracias al modelo del sistema, a través de su capa de abstracción de hardware. Esto incluye tanto a programas que son ejecutados desde la línea de comandos hasta los más sofisticados programas con interfaces gráficas.	No hay mucho que decir en este aspecto tocante a Windows NT. Este sistema ha evolucionado basado en el diseño básico de la PC, en que cada computadora tiene una pantalla, un teclado y un mouse. No existe en este aspecto opción alguna más que utilizar programas cliente-servidor para la administración remota. Y si bien esto ha sido llevado a cabo para varios de los programas del sistema (por ejemplo, el Administrador de usuarios para dominios), tiene que ser implementado individualmente a cada uno de estos, proceso tedioso, ineficiente y rara vez llevado a cabo. Aunque en el último año han aparecido varios programas que permiten utilizar remotamente una computadora con Windows, como el VNC (Virtual Network Computing, software libre desarrollado por AT&T) o el Citrix Winframe, estos dan un desempeño demasiado bajo o requieren demasiados recursos del servidor como para ser considerados prácticos.

Figura 1: Usuarios de Linux en el mundo desde 1994



Además de esta gráfica se hace referencia de algunas empresas que utilizan Unix o cualquiera de sus versiones como Linux, Red Hat, PPP. Mandrake, etc, y sus entornos gráficos.

Amazon.com Digital UNIX Alpha Server 2000.

Boeing. HP-UX, IRIX, Solaris y algo de NT.

Dallas Cowboys. IRIX y UNIX System V Rel. 4.0.

Dow Corning. Solaris.

Hotmail (Pertenece a Microsoft). Solaris, FreeBSD. Intentaron migrar a NT, pero ofrecer los servicios a más de 10, 000,000 de usuarios, fué demasiado para NT.

Servicio Postal de Estados Unidos. Cuentan con 900 sistemas Linux a lo largo del E.U. para determinar automáticamente el destino de la correspondencia. Cada sistema consiste de 5 Pentium Pro dual 200MHz o un sistema dual sencillo.

Yahoo!. FreeBSD.

IBM. Puso a disposición una línea de servidores de última generación con Linux y planea masificar Linux en PCs de escritorio dentro de un par de años.

Si bien aquí se demuestran las ventajas de los sistemas basados en Unix, como es Linux, sobre las plataformas de Microsoft mejor conocidas como Windows no es el objetivo de este trabajo, puesto que como se manifestó anteriormente, dependiendo de los requerimientos se escogerá el sistema operativo a utilizar y debemos tener el más optimo respecto al trabajo a

desempeñar, aunque se recomienda ampliamente tener una plataforma más segura como lo es Linux o cualquiera basada en UNIX.

Así mismo, a través de esta investigación nos damos cuenta que Linux esta empujando cada vez más, a que cualquier Técnico o Ingeniero que entra al mundo de las telecomunicaciones tenga al menos los conocimientos básicos de este sistema operativo.

En resumen Linux tiene todo y más que Windows para desempeñar un buen trabajo en ambientes de interoperabilidad, por lo que al tener conocimiento de esta plataforma amplía el campo de trabajo en el que nos podemos desempeñar como Ingenieros.

CAPITULO II
INSTALACION

4. Instalación de LINUX

4.1 Preparación para instalar Linux

Una vez que ha obtenido una distribución de Linux, para éste caso práctico la distribución fue REDHAT; estamos preparados para instalar el sistema. Esto supone cierto grado de planificación, sobre todo si en el computador se usan actualmente otros sistemas operativos. En las secciones siguientes verá como preparar la instalación de Linux.

4.2 Visión general de la instalación

A pesar de ser diferente cada distribución de Linux, el método utilizado para instalar el software es, en general, como sigue:

a) Reparticionamiento disco(s) duro(s). Si tiene instalados otros sistemas operativos, necesita reparticionar los discos con el fin de reservar espacio para Linux.

b) Arranque de la instalación de Linux. Cada distribución de Linux incluye algo para arrancar inicialmente e instalar el software, usualmente un disquete de arranque. Arrancando de esta forma, entrará en un programa de instalación para el resto del software, o bien le permitirá seguir instalándolo a mano.

c) Creando las particiones para Linux. Después de reparticionar el disco para reservar espacio para Linux, debe crear particiones de Linux en dicho espacio. Esto se realiza con el programa fdisk.

d) Creando los sistemas de ficheros y el espacio de intercambio. En este momento, se debe crear uno o más sistemas de ficheros, utilizados para guardar los archivos, en las particiones recién creadas. Además, si piensa usar espacio de intercambio ("swap"), debe crear dicho espacio en una de las particiones para Linux.

e) Instalando los programas en los sistemas de ficheros. Finalmente, debe instalar el software en los nuevos sistemas de ficheros.

La mayoría de las distribuciones de Linux proporcionan un programa de instalación que le guiará en cada paso de la instalación, y automatiza algunos de esos pasos. Es necesario tener en cuenta que cualquiera de los siguientes pasos pueden estar automatizados o no, dependiendo de la distribución.

La distribución RedHat de Linux, sólo requiere que se reparticione el disco, utilizando fdisk y el programa setup para completar los restantes pasos.

4.3 Conceptos sobre particiones

En general, los discos duros se encuentran divididos en particiones, donde cada partición corresponde a un sistema operativo. Por ejemplo, en un disco duro se pueden tener varias particiones, una dedicada a MS-DOS, otra a OS/2 y otra a Linux.

Si tiene ya otro software instalado en el sistema, es probable que deba cambiar el tamaño de las particiones con el fin de reservar espacio para Linux. En el espacio reservado se crearán una o más particiones para almacenar el software de Linux y el espacio de intercambio. A este proceso se le llama **reparticionar**.

La mayoría de los sistemas MS-DOS utilizan una única partición que ocupa todo el disco. Para MS-DOS, esta partición es accedida como C:. Si hay más de una partición, MS-DOS las llamara D:, E:, y así sucesivamente, de modo que cada partición actúa como si fuera un disco duro independiente.

En el primer sector del disco está el registro de arranque maestro junto a la tabla de particiones. El registro de arranque (como su nombre indica) se usa para arrancar el sistema. La tabla de particiones contiene información acerca del lugar y el tamaño de cada partición.

Hay tres clases de particiones: *primarias*, *extendidas*, y *lógicas*. De estas, las más usadas son las primarias. Sin embargo, debido al límite del tamaño de la tabla de particiones, solo pueden tenerse hasta cuatro particiones primarias en un disco.

La forma de superar este límite de cuatro particiones es usar particiones extendidas. Una partición extendida no tiene datos en ella misma; en su lugar, actúa como "soporte" de particiones lógicas. Por lo tanto, se puede crear una partición extendida que ocupe todo el disco, y dentro crear varias particiones lógicas. Sin embargo, sólo puede tener una partición extendida por disco.

4.4 Necesidades de reparticionado en Linux

Antes de que se explique como reparticionar los discos, es necesario que tenga una idea acerca del espacio que requiere disponer para la instalación de Linux.

En los sistemas UNIX, los ficheros se almacenan en un sistema de ficheros, que es esencialmente una zona del disco duro (u otro dispositivo, como un CD-ROM o un disquete) formateado para almacenar ficheros. Cada sistema de ficheros se encuentra asociado con una parte especifica del árbol de directorios; por ejemplo, en la mayoría de los sistemas, existe un sistema de ficheros para todos los ficheros del directorio */usr*, otro para */tmp*, etc. El sistema de ficheros raíz es el principal, que corresponde al directorio raíz, /.

Bajo Linux, cada sistema de ficheros ocupa una partición del disco duro. Por ejemplo, si hay un sistema de ficheros para / y otro para */usr*, son necesarias dos particiones para almacenar ambos sistemas.

Antes de instalar Linux, necesitará preparar sistemas de ficheros para almacenar el software de Linux. Por lo menos debe tener un sistema de ficheros (el sistema de ficheros raíz), y una partición reservada a Linux. La mayoría de los usuarios de Linux optan por almacenar todos sus ficheros en el sistema de ficheros raíz, pues en la mayor parte de los casos es más fácil de gestionar que tener diferentes sistemas de ficheros y particiones.

Sin embargo, se pueden crear varios sistemas de ficheros para Linux, por ejemplo, se pueden usar sistemas separados para */usr* y */home*.

¿Por que usar más de un sistema de ficheros? Lo más habitual es por seguridad; si por alguna razón, uno de sus sistemas de ficheros resulta dañado, los otros normalmente no resultarán afectados. Por otro lado, si se almacenan todos sus ficheros en el sistema de

ficheros raíz, y por alguna razón resulta dañado, se pueden perder todos los ficheros de una vez. Sin embargo, esto no es lo habitual; si el administrador hace copias de seguridad (backups) regularmente.

Otra razón para utilizar varios sistemas de ficheros es repartir el almacenamiento entre varios discos duros. Si existen, 40 megabytes libres en un disco duro y 50 en otro, es posible crear un sistema de ficheros raíz de 40 megabytes en el primer disco y un sistema */usr* de 50 megabytes en el otro. Actualmente no es posible que un sistema de ficheros abarque varios discos; si el espacio libre de disco esta repartido entre los discos, se debe utilizar varios sistemas de ficheros para aprovecharlos.

En resumen, Linux requiere por lo menos una partición, para el sistema de ficheros raíz. Si se desea crear varios sistemas de ficheros, se necesitará una partición por cada sistema de ficheros.

Algunas distribuciones de Linux crean particiones y sistemas de ficheros de forma automática, de modo que no es necesario crearlos. Otro asunto a considerar cuando se deciden las particiones, es el espacio de intercambio (swap). Si se desea usar espacio de intercambio en Linux, se tienen dos opciones. La primera es usar un fichero de intercambio que existe dentro de uno de los sistemas de ficheros de Linux. Se crea el fichero de intercambio para usarlo como RAM virtual una vez instalado el software. La segunda opción es crear una partición de intercambio, una partición reservada exclusivamente como espacio de swap. La mayoría de la gente usa una partición para el intercambio en lugar de un fichero.

Por lo general, se crearán dos particiones para Linux: una para ser usada como sistema de ficheros raíz, y la otra como espacio de intercambio. Por supuesto, hay otras opciones pero ésta es la básica. El espacio de swap no es obligatorio en Linux, pero está muy recomendado si se posee menos de 16 megabytes de memoria física lo cual es bastante lógico en este caso, ya que se supone que estamos trabajando con el mínimo de Hardware.

También se necesita conocer el espacio requerido para cada partición. El tamaño de los sistemas de ficheros de Linux depende en gran parte de qué software se requiere instalar en él, y de la distribución de Linux que esté utilizando. Un sistema pequeño puede utilizar sólo 80 megabytes o menos, un sistema grande siempre necesitará 100 a 150 megabytes, o más. Teniendo en cuenta que hay que añadir a esto el espacio extra para los directorios de usuario, expansiones futuras, etc.

El tamaño de las particiones de swap (debe elegirse una para esto) depende de la RAM virtual que se necesite. Lo típico es crear una partición de intercambio del doble de espacio de la RAM física; por ejemplo, si tiene 32 megabytes de RAM, una partición de 64 megabytes suele bastar. Por supuesto, esto es sólo una idea; la cantidad de espacio de swap que requiere dependerá del software que necesita ejecutar. Si se tiene una gran cantidad de memoria física (digamos, 128 megabytes o más) puede que al final no se necesite espacio de intercambio.

4.5 Repartición de los discos

En esta sección se descubrirá como cambiar el tamaño de las particiones actuales (si las hay) para reservar espacio para Linux. Si está instalando Linux en un disco duro "limpio", puede obviar esta sección.

La manera habitual de cambiar el tamaño de una partición es borrarla (lo que implica borrar toda la información que contenga) y rehacerla. Antes de reparticionar los discos, hay que hacer un backup. Después de cambiar las particiones, puede proceder a reinstalar el software desde el backup. Sin embargo, puede encontrar programas para MS-DOS que consiguen cambiar el tamaño de las particiones de forma no destructiva. Uno de estos se conoce como "FIPS", y puede encontrarse en muchos servidores de FTP de Linux.

Se debe tener presente que debido a que se empequeñecen las particiones originales, no se va a poder reinstalar todo el software que se tenía antes. En este caso, hay que borrar el software innecesario para permitir que el resto quepa en las particiones más pequeñas.

El programa utilizado para hacer particiones es fdisk. Cada sistema operativo tiene su propia versión de este programa; por ejemplo, bajo MS-DOS, se activa con el comando FDISK. Es necesario consultar la documentación de los sistemas operativos en uso para obtener información sobre este asunto. Aquí se referirá únicamente a MS-DOS con FDISK, pero lo que puede ser fácilmente extrapolado a otros sistemas operativos.

Es aconsejable no modificar o crear particiones para otros sistemas operativos (incluyendo Linux) utilizando FDISK bajo MS-DOS. Sólo pueden modificarse particiones de cada sistema operativo con la versión de fdisk correspondiente a ese sistema; para éste trabajo, se crearán las particiones para Linux utilizando el programa fdisk que viene con Linux. Más adelante, se discutirá como crear particiones de Linux, pero por ahora, el objetivo será cambiar el tamaño de las actuales.

Suponga que se tiene un solo disco duro en el sistema, dedicado, por ahora, enteramente a MS-DOS. El disco duro contiene una partición MS-DOS, conocida habitualmente como "C:". Puesto que este método de reparticionado destruirá todos los datos de la partición, será necesario crear un disco de sistema MS-DOS "arrancable" que contenga lo necesario para ejecutar FDISK y restaurar el software desde el backup cuando se complete el proceso de reparticionado.

En muchos casos, se pueden usar para esto los discos de instalación de MS-DOS. Sin embargo, si necesita el disco de sistema, debe formatearlo mediante el comando

FORMAT /s A:

Con esto, se copian en el disquete todas las utilidades de MS-DOS necesarias (normalmente, casi todo lo que hay en el directorio \DOS de el disco), así como los programas *FORMAT.COM* y *FDISK.EXE*. Con esto, se puede arrancar desde éste disquete, y ejecutar el comando

FDISK C:

La utilización de FDISK debería ser auto explicativa, de todas maneras, existe documentación de MS-DOS para obtener detalles. Cuando comience el programa FDISK, deberá seleccionar el menú de opciones para mostrar la tabla de particiones, y anotar la información que se le muestre. Es importante guardar copia de la configuración original en caso de que se deba detener la instalación de Linux.

Para borrar una partición, se selecciona la opción del menú: "*Delete an MS-DOS Partition or Logical DOS Drive*" (Eliminar partición o unidad lógica DOS). Se especifica el tipo de partición que se va a borrar (primaria, extendida o lógica) y el número de la partición.

Para crear una nueva partición para MS-DOS (más pequeña), se selecciona la opción de fdisk: "*1. Crear partición DOS o unidad lógica DOS*". Se especifica el tipo de partición (primaria, extendida o lógica) y el tamaño (en megabytes).

Después de hacer esto mediante FDISK, se abandona el programa y se reformatea las nuevas particiones. Por ejemplo, si se cambió el tamaño de la partición C: se tecleará el comando:

FORMAT /s C:

Ahora se puede reinstalar el software desde el backup.

4.6 Instalación del software de Linux

Después de modificar las particiones para reservar espacio a Linux, se dan las condiciones para instalar el software. A continuación se muestra un resumen del procedimiento a seguir:

a) Arrancar con el dispositivo de arranque de Linux (disquete);
b) Ejecutar fdisk bajo Linux para crear las particiones de Linux;
c) Ejecutar *mke2fs* y *mkswap* para crear los sistemas de ficheros y el espacio de intercambio;
d) Instalar el software de Linux;
e) Y finalmente, instalar el cargador LILO en el disco duro, o crear un disco de arranque con el fin de ejecutar Linux.

Uno o más pasos de los anteriores pueden estar automatizados por los programas de instalación, según la distribución que este utilizando. Es necesario consultar la documentación de la distribución para ver las instrucciones específicas.

4.7 Arranque de Linux

El primer paso es iniciar el computador con el dispositivo de arranque de Linux, que suele ser un disco "*boot*" que contiene un pequeño sistema Linux. Tras arrancar con el disquete, se presentará un menú de instalación que le guiará en el proceso de instalación. En otras distribuciones, se le mostrará un prompt de login cuando arranque. Aquí se suele entrar como *root* o *install* para comenzar el proceso de instalación. La documentación que viene con cada distribución explicará que se necesita para arrancar Linux.

La mayoría de las distribuciones de Linux utilizan un disquete de arranque, o CD de arranque que permite introducir parámetros del hardware durante el tiempo de arranque, para

forzar la detección de los dispositivos. Por ejemplo, si una controladora SCSI no se detecta durante el arranque normal, es necesario rearrancar y especificar los parámetros del hardware (como direcciones E/S e IRQ) en el prompt de arranque.

Asimismo, las máquinas PS/1, ThinkPad y ValuePoint de IBM no almacenan la información de geometría de los discos en la CMOS, con lo que debe especificarla durante el arranque.

El prompt de arranque se muestra siempre que se arranca con el disquete o CD. Este es el caso de la distribución RedHat. En otras, es necesario mantener pulsadas las teclas *shift* o *ctrl* mientras se arranca. En pantalla aparecerá la palabra boot:, y tal vez otros mensajes.

Para arrancar sin más parámetros especiales, deberá pulsar *enter* en el prompt del arranque, se debe estar atento a los mensajes del arranque. Si se tiene una controladora SCSI, se verá una lista de hosts SCSI detectados. Si aparece el mensaje:

SCSI: 0 hosts

Es porque no se detectó la controladora SCSI, y tendrá que seguir el siguiente procedimiento.

Además, el sistema informará de las particiones y dispositivos detectados. Si cualquier parte de esta información es incorrecta, se tendrá que forzar la detección del hardware. Por otro lado, si todo el hardware es correctamente detectado, podrá pasar a la siguiente sección.

Para forzar la detección del hardware, deberá ingresar los parámetros adecuados en el prompt de arranque, utilizando la siguiente sintaxis:

ramdisk <parameters. . . >

Hay cierto número de parámetros disponibles; aquí se mostrarán los más comunes.

hd=<cylinders>,<heads>,<sectors>

Especifica la geometría del disco, requerido para sistemas como el IBM PS/1, ValuePoint y ThinkPad. Por ejemplo, si el disco tiene 683 cilindros, 16 cabezas y 32 sectores por pista, se introduce: *ramdisk hd=683, 16,32*

tmc8xx=<memaddr>,<irq>

Se especifican las direcciones e IRQ para el controlador SCSI Future Domain TMC-8xx. Por ejemplo: *ramdisk tmc8xx=0xca000,5*

El prefijo 0x debe utilizarse para todos los valores que se dan en hexadecimal. Esto se cumple con todas las opciones siguientes.

st0x=<memaddr>,<irq>

Especifica las direcciones e IRQ para el controlador Seagate ST02.

t128=<memaddr>,<irq>

Especifica las direcciones e IRQ para el controlador Trantor T128B.

ncr5380=<port>,<irq>,<dma>

Especifica el puerto, IRQ y canal DMA para el controlador genérico NCR5380.

aha152x=<port>,<irq>,<scsi_id>,1

Especifica puerto, IRQ e identificador SCSI para controladores AIC-6260. Esto incluye a los controladores Adaptec 1510, 152x y Soundblaster-SCSI.

Para cada uno de estos, se debe ingresar *ramdisk* seguido del parámetro a utilizar. Si existen dudas acerca de estas opciones de arranque, podrá leer el documento Linux SCSI HOWTO, que debe estar disponible en cualquier FTP-site de Linux, así como el documento Linux CD-ROM HOWTO. Estos documentos describen temas de compatibilidad de hardware con mucho más detalle.

5. Dispositivos y particiones en Linux

Como ya se mencionó, muchas distribuciones necesitan que se creen a mano las particiones de Linux utilizando el programa fdisk. Otras pueden ser creadas automáticamente. En cualquier caso, debe conocer los nombres para los dispositivos y las particiones en Linux.

Bajo Linux, los dispositivos y las particiones tienen nombres muy distintos a los utilizados en otros sistemas operativos. Bajo MS-DOS, las disqueteras se identifican como A: y B:, mientras que las particiones del disco duro se identifican como C:, D, etc. Bajo Linux, la denominación es algo diferente.

Los controladores de dispositivos, que se encuentran en el directorio */dev*, se usan para comunicarse con los dispositivos del sistema (como discos duros o ratones). Por ejemplo, si se posee un ratón, se le puede acceder a través del controlador */dev/mouse*. Las disqueteras, discos duros y particiones tienen cada uno un controlador propio.

Tabla 3: lista los nombres de diversos controladores.

Dispositivo	Nombre
Primera disquetera (A:)	/dev/fd0
Segunda disquetera (B:)	/dev/fd1
Primer disco duro (todo el disco)	/dev/hda
Primer disco duro, partición primaria 1	/dev/hda1
Primer disco duro, partición primaria 2	/dev/hda2
Primer disco duro, partición primaria 3	/dev/hda3
Primer disco duro, partición primaria 4	/dev/hda4
Primer disco duro, partición lógica 1	/dev/hda5
Primer disco duro, partición lógica 2	/dev/hda6
..	
.	
Segundo disco duro (todo el disco)	/dev/hdb
Segundo disco duro, partición primaria 1	/dev/hdb1
..	
.	
Primer disco duro SCSI (todo el disco)	/dev/sda
Primer disco duro SCSI, partición primaria 1	/dev/sda1
..	
Segundo disco duro SCSI (todo el disco)	/dev/sdb
Segundo disco duro SCSI, partición primaria 1	/dev/sdb1

Además, los discos duros SCSI se nombran de manera diferente a otros discos. Los IDE, MFM y RLL se acceden a través de los dispositivos */dev/hda*, */dev/hdb*, etc. Las particiones de */dev/hda* son */dev/hda1*, */dev/hda2*, etc. Sin embargo, los dispositivos SCSI se nombran como */dev/sda*, */dev/sdb*, etc., y las particiones como */dev/sda1*, */dev/sda2*, etc. Por ejemplo, suponga que se tiene un disco duro IDE con 3 particiones primarias. Las dos primeras son para MS-DOS, y la tercera es extendida y contiene dos particiones lógicas, ambas para ser usadas con Linux. Los dispositivos quedarán representados con:

Tabla 4: Ejemplo particiones en Linux

Primera partición MS-DOS (C:)	/dev/hda1
Segunda partición MS-DOS (D:)	/dev/hda2
Partición extendida	/dev/hda3
Primera partición lógica de Linux	/dev/hda5
Segunda partición lógica de Linux	/dev/hda6

Observe que se ha saltado */dev/hda4*, ya que corresponde a la cuarta partición primaria, que no existe en el ejemplo. Las particiones lógicas se nombran de forma consecutiva partiendo de */dev/hda5*.

5.1 Creación de las particiones en Linux

Ahora podrá crear las particiones de Linux con el comando fdisk. Como se explicó anteriormente, va a tener que crear, al menos, una partición para el software de Linux propiamente tal y otra para el área de intercambio.

Después de arrancar el disquete, se ejecuta el comando fdisk tecleando:

fdisk <drive>;

Donde <drive> es el nombre de dispositivo con el que Linux identifica el disco duro donde quiere realizar las particiones (véase la Tabla N° 1). Por ejemplo, si desea ejecutar fdisk sobre el primer disco SCSI del sistema, se utiliza el comando fdisk */dev/sda*. Por defecto, fdisk actúa sobre */dev/hda* (el primer disco IDE).

Para crear particiones de Linux en más de un disco, se ejecuta fdisk una vez por disco.

fdisk /dev/hda

Command (m for help):

En este punto, fdisk está esperando un comando; puede teclear m para obtener una lista de opciones.

Command (m for help): m
Command action
a toggle a bootable flag
d delete a partition
l list known partition types
m print this menú
n add a new partition
p print the partition table
q quit without saving changes
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

Command (m for help):

El comando *n* se usa para crear una nueva partición. Para salir de fdisk sin salvar cambios, se utiliza el comando *q*. Para salir escribiendo los cambios en la tabla de particiones, se utiliza el comando *w*.

Lo primero que se debe hacer es mostrar la tabla de particiones actual y anotar sus datos, para referencias posteriores. Para esto, se usa el comando *p*.

Command (m for help): *p*
Disk /dev/hda: 16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes

Device	Boot	Begin	Start	End	Blocks	Id	System
/dev/hda1	*	1	1	203	61693	6	DOS 16-bit >=32M

Command (m for help):

En este ejemplo, se tiene una partición única en */dev/hda1*, con 61693 bloques (aproximadamente 60 megabytes).[4] Esta partición comienza en el cilindro 1 y finaliza en el 203, en total el disco tiene 683 cilindros de los cuales 480 están libres para crear particiones de Linux. Para crear una nueva partición, se utiliza el comando *n*. En este ejemplo se crean dos particiones primarias (*/dev/hda2* y */dev/hda3*) para Linux.

Command (m for help): *n*
Command action
e extended
p primary partition (1-4)
p

Aquí, fdisk pide el tipo de partición a crear: extendida o primaria. En este ejemplo se elige *p*, pues sólo se van a crear particiones primarias.

Partition number (1-4):

fdisk preguntara entonces por el número de la partición a crear; puesto que la *1* esta en uso, la primera partición para Linux debe ser la *2*.

Partition number (1-4): *2*
First cylinder (204-683):

[4] En Linux, un bloque son 1024 bytes.

Ahora deberá ingresar el cilindro de comienzo de la partición. Dado que actualmente no están en uso los cilindros 204 a 683, escogerá el primero disponible (204), ya que no hay razón para dejar particiones vacías.

First cylinder (204-683): 204
Last cylinder or +size or +sizeM or +sizeK (204-683):

Ahora fdisk preguntará acerca del tamaño de la partición a crear, puede hacerlo especificando el cilindro de terminación de la partición o introduciendo directamente el tamaño requerido, en bytes, kilobytes, o megabytes, como es necesario que la partición ocupe 80 megabytes, se especifica como: *+80M*. Cuando se indica el tamaño de esta forma, fdisk lo redondea a un número de cilindros.

Last cylinder or +size or +sizeM or +sizeK (204-683): *+80M*

Warning: Linux cannot currently use 33090 sectors of this partition

Si aparece un mensaje como el anterior, se puede ignorar. Fdisk imprime este aviso debido a que es un programa antiguo que data de cuando las particiones de Linux no podían superar los 64 megabytes.

Luego puede pasar a crear la segunda partición. Como ejemplo, se creará una de 10 megabytes.

Command (m for help): *n*
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): *3*
First cylinder (474-683): *474*
Last cylinder or +size or +sizeM or +sizeK (474-683): *+10M*

Finalmente, vea la tabla de particiones, una vez más, debe anotar la información que se presenta sobre todo los tamaños en bloques de las nuevas particiones. Necesita conocerlos cuando tenga que crear, más tarde, los sistemas de ficheros. Además, debe verificar que las particiones no se solapen.

```
Command (m for help):  p

Disk /dev/hda:  16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes

 Device Boot   Begin    Start      End   Blocks   Id   System
/dev/hda1    *       1        1     203    61693    6   DOS 16-bit >=32M
/dev/hda2          204      204     473    82080   81   Linux/MINIX
/dev/hda3          474      474     507    10336   81   Linux/MINIX
```

Como se puede ver, ahora en */dev/hda2* tiene una partición de 82080 bloques (aproximadamente 80 megabytes) y en */dev/hda3* tiene 10336 bloques (unos 10 megabytes).

Existen muchas distribuciones (como la Slackware) que requieren que se utilice el comando *t* en fdisk para cambiar el tipo de la partición elegida para el intercambio a "Linux swap", normalmente con el numero 82. Se puede usar el comando *L* para ver una lista de tipos de particiones conocidas, y luego *t* para establecer el tipo de la partición de intercambio a "Linux swap".

De esta forma, el software de instalación podrá encontrar automáticamente sus particiones de swap en función del tipo, si el software de instalación no pudiera reconocer la partición de intercambio, debe repetir la ejecución de fdisk y utilizar el comando *t* sobre la partición en cuestión.

En el ejemplo anterior, los cilindros sobrantes (508 a 683) se quedan sin usar. Puede que desee hacerlo así, para más adelante crear más particiones.

Finalmente, se utiliza el comando *w* para escribir los cambios en el disco y salir.

```
Command (m for help):  w
#
```

Es bueno recordar que ningún cambio hecho durante la ejecución de fdisk tendrá efecto hasta que se teclee el comando *w*, por lo que se pueden probar diferentes configuraciones y salvarla sólo cuando esté decidido, además, se puede usar el comando *q* para abandonar fdisk sin hacer ningún cambio. Es necesario tener en mente que las particiones de otros sistemas operativos no deben tocarse desde el programa fdisk de Linux.

Asimismo, hay que recordar que no se puede arrancar Linux desde una partición que comience más allá del cilindro 1023. Por lo tanto, se puede crear la partición de raíz en el rango inferior a este cilindro o, si esto es imposible, arrancar siempre desde un disquete.

Algunas distribuciones de Linux necesitan rearrancar el sistema tras ejecutar fdisk. Esto permite que los cambios en la tabla de particiones tengan efecto. Las nuevas versiones de fdisk cambian de forma automática esta información en el núcleo. Lo más seguro es volver a arrancar tras crear las particiones.

5.2 Creación del espacio de intercambio (swap)

Si está pensando en usar una partición de intercambio para obtener RAM virtual, es ahora el momento de hacerlo.[5]

En el la sección siguiente se discutirá la preparación de un fichero de swap para el caso de que no desee usar una partición para esto.

Muchas distribuciones necesitan que se cree y active la partición de intercambio antes de instalar el software. Si se dispone de poca RAM física, la instalación puede no ir bien, a menos que se active una cierta cantidad de swap.

La distribución Redhat necesita que se cree el área de swap antes de la instalación, si se tienen 4 megabytes o menos. Si este no es el caso, el procedimiento de instalación de Redhat (y otras distribuciones) puede usarse para preparar la partición de intercambio automáticamente. Si está seguro, es mejor seguir con el procedimiento descrito aquí.

El comando utilizado para preparar una partición de intercambio es *mkswap*, tecleándose

[5] Algunas distribuciones de Linux preparan el área de intercambio automáticamente, o bien mediante un menú de instalación.

mkswap -c <partition> <size>

Donde <*partition*> es el nombre de la partición de swap y <*size*> es el tamaño de la partición, en bloques.[6] Por ejemplo, la partición de intercambio es */dev/hda3* y tiene 10336 bloques, se teclea el comando:

 # mkswap -c /dev/hda3 10336

La opción *-c* indica a mkswap que compruebe si hay bloques erróneos en la partición mientras la crea.

Si se usan varias particiones de intercambio, se necesitará ejecutar el comando *mkswap* apropiado para cada partición.

Después de preparar el área de swap, hay que decirle al sistema que la use. Normalmente, el sistema comienza a usarla automáticamente durante el arranque, sin embargo, como aun no tiene instalado el software de Linux, tendrá que activarla a mano.

El comando para hacerlo es *swapon*, y tiene el formato:

swapon <partition>

En el ejemplo anterior, para activar el espacio de intercambio en */dev/hda3*, se usará el comando:

 # swapon /dev/hda3

[6] Este tamaño es dado por fdisk, utilizando la opción p. Un bloque en Linux son 1024 bytes.

6. Creación de los sistemas de ficheros

Antes de que se puedan usar las particiones de Linux para almacenar ficheros, hay que crear los sistemas de ficheros en ellas. La creación de un sistema de ficheros es análoga a formatear una partición de MS-DOS u otros sistemas operativos.

Hay varios tipos de sistemas de ficheros disponibles en Linux, cada tipo de sistema de ficheros tiene su propio formato y características (como longitud del nombre de los ficheros, tamaño máximo, etc). Además, Linux soporta sistemas de ficheros "de terceros" como el de MS-DOS.

El tipo de sistema de ficheros más usado es el Sistema de Ficheros Extendido 2, o *ext2fs*. El *ext2fs* es uno de los más eficientes y flexibles sistemas; permite hasta 256 caracteres en los nombres de los ficheros y tamaños de estos de hasta 4 Terabytes. De momento se analizará solo el sistema *ext2fs*.

Si está instalando la distribución Redhat, el propio procedimiento de instalación normal creará los sistemas de ficheros de forma automática. Si desea crear sus propios sistemas a mano, es necesario seguir el método que a continuación se describe.

Para crear un sistema tipo *ext2fs* se utiliza el comando:

mke2fs -c <partition> <size>

Donde <*partition*> es el nombre de la partición, y <*size*> es el tamaño de la partición en bloques, por ejemplo, para crear un sistema de 82080 bloques en /dev/hda2, se usa el comando

 # mke2fs -c /dev/hda2 82080

Si quiere usar varios sistemas de ficheros en Linux, necesita repetir el comando *mke2fs* por cada sistema de ficheros.

7. Instalación del software

Finalmente, ya está preparado para instalar el software en el sistema. Cada distribución tiene una forma distinta de hacerlo. Muchas tienen un programa que guía paso a paso este proceso, en otras, es necesario montar los sistemas de ficheros en un directorio (como */tmp*) y copiar el software dentro de éste a mano. En las distribuciones en CD-ROM es posible seguir la opción de instalar una parte de lo que contiene en el disco duro y dejar el resto (la mayor parte) en el CD-ROM.

Algunas distribuciones ofrecen diversos mecanismos para instalar el software. Por ejemplo, es posible instalarlo directamente desde una partición MS-DOS del disco duro, en lugar de hacerlo desde los disquetes, o incluso hacerlo a través de una red TCP/IP mediante FTP o NFS.

Por ejemplo, las antiguas distribuciones necesitaban que se creen las particiones con fdisk, y el espacio de intercambio con *mkswap* y *swapon* (si se tenía 4 megabytes o menos de RAM), y a continuación se ejecutaba el programa setup, que guiaba todo el proceso mediante un menú bastante auto explicativo.

7.1 Creación del disco de arranque o instalación del LILO

Cada distribución proporciona mecanismos para arrancar Linux cuando ya esté instalado en el sistema. En la mayoría de los casos se creará un disquete "boot" que contiene el núcleo de Linux configurado para usar con el recién creado sistema de ficheros raíz. Para arrancar Linux, se debe hacer desde ese disquete y tras el arranque se pasa el control al disco duro. En otras distribuciones, el disco de arranque es el propio disquete de instalación.

La mayoría de las distribuciones dan la opción de instalar LILO en el disco duro. LILO es un programa que se instala en el registro maestro de arranque del disco, y está preparado para arrancar varios sistemas operativos, entre los que se incluyen MS-DOS y Linux, permitiendo elegir con que sistema quiere arrancar en cada momento.

Con el fin de instalar LILO correctamente, es necesario conocer bastante información acerca de la configuración del disco, por ejemplo, que particiones contiene cierto sistema operativo, como arrancar cada sistema operativo, etc. En la mayoría de las distribuciones,

cuando se instala LILO, tratan de elegir la mejor configuración para este. Aunque no es lo habitual, la instalación automatizada de LILO puede fallar, dejando el registro de arranque maestro de su disco inservible (aunque es difícil que esto llegue a hacerle perder datos del disco).

En muchos casos, lo mejor es usar un disquete de arranque, hasta que esté en condiciones de configurar LILO manualmente.

Hoy en día las versiones actuales de Linux, permiten realizar todo el proceso de instalación en forma auto explicativa, de la misma manera que si instala Windows, partiendo en forma directa del cdrom, y desde ahí comienza con la creación de las particiones, instalación de la Swap y disposición de los sistemas de ficheros; es más, en las últimas versiones de RedHat, existe la posibilidad de seleccionar tipos de instalaciones como Servidor, Workstation e inclusive para equipos portátiles Laptops. La diferencia entre cada uno son los paquetes que se van colocando en el sistema, por ejemplo, para la instalación del servidor, se incluyen los programas *Apache*, *sendmail*, *ftp server*, control de usuarios, entre otros, los cuales no son instalados en la modalidad Workstation.

La forma exacta para instalar el software de Linux difiere en gran parte según la distribución.

En el siguiente capítulo se describirá los puntos de operación más habituales para comenzar a usar Linux.

CAPITULO III
TRABAJANDO CON LINUX

8. Entorno de Usuarios

LINUX es un sistema operativo multitarea y multiusuario. Esto significa que puede haber más de una persona usando un computador a la vez, cada uno de ellos ejecutando diferentes aplicaciones. (Esto difiere de MS-DOS, donde sólo una persona puede usar el sistema en un momento dado). Bajo LINUX, para que los usuarios puedan identificarse en el sistema, deben presentarse, proceso que consta de dos pasos: Introducir el nombre de usuario (*login*) (el nombre con que será identificado por el sistema), y una palabra de paso (*password*), la cual es la llave personal secreta para entrar en la cuenta. En los sistemas LINUX tradicionales, el administrador del sistema asignará el nombre de usuario y una palabra de paso inicial en el momento de crear la cuenta de usuario. Si usted es el administrador del sistema, debe configurar su propia cuenta antes de poder presentarse, más adelante se verá como hacerlo. Para el resto de las discusiones, se usará el nombre de usuario "**Admin**".

Además, cada sistema LINUX tiene un nombre del sistema (*hostname*) asignado. Este "*hostname*" le da el nombre a la máquina, además de carácter y “personalidad”. El nombre del sistema es usado para identificar máquinas en una red, pero incluso aunque la máquina no esté en red, debería tener su nombre, tema que también se tratará más adelante. En los ejemplos, el nombre del sistema será "**Telsur**"

8.1 Creación de una cuenta

Antes de poder usar el sistema, deberá configurar una cuenta de usuario. Esto es necesario, porque no es buena idea utilizar la cuenta de root para los usos normales. La cuenta de root debería reservarse para el uso de comandos privilegiados y para el mantenimiento del sistema. Para crear su propia cuenta, necesitará entrar en la cuenta de root y usar las órdenes *useradd* o *adduser*.

En el momento de presentarse en el sistema, verá la siguiente línea de comandos en la pantalla:

telsur login:

Ahora, introducirá el nombre de usuario:

telsur login: admin
Password:

Ahora introducirá la palabra de paso. Esta no será mostrada en la pantalla conforme se va tecleando, por lo que se debe teclear cuidadosamente. Si se introduce una palabra de paso incorrecta, se mostrará el siguiente mensaje

Login incorrect

y tendrá que intentarlo nuevamente.

Una vez que se ha introducido correctamente el nombre de usuario y la palabra de paso, esta oficialmente "presentado" en el sistema y libre para comenzar a trabajar.

8.2 Consolas virtuales

La consola del sistema es el monitor y teclado conectado directamente al sistema. (Como UNIX es un sistema operativo multiusuario, puede tener otros terminales conectados a puertos serie del sistema, pero estos no serán la consola). Linux, como otras versiones de UNIX, proporciona acceso a consolas virtuales (o VC's), las cuales le permitirán tener más de una sesión de trabajo activa desde la consola a la vez.

Para demostrar esto, entré al sistema (como se vio anteriormente). Ahora pulse *alt-F2*, debería ver la pregunta login: de nuevo; está viendo la segunda consola virtual ha entrado en el sistema por la primera. Para volver a la primera VC, pulse *alt-F1*.

Un sistema Linux recién instalado probablemente le permitirá acceder a las primeras cuatro VC's, usando *alt-F1* a *alt-F4*, pero es posible habilitar hasta 12 VC's una por cada tecla

de función del teclado. Como se puede ver, el uso de VC's es muy potente, puede estar trabajando en diferentes VC's a la vez.

Mientras que el uso de VC's es algo limitado (después de todo, sólo puede mirar un VC cada vez), esto debería darle una idea de las capacidades multiusuario del sistema. Mientras está trabajando en el VC #1, puede conmutar al VC #2 y comenzar a trabajar en otra cosa.

8.3 Interpretes de comandos y comandos

En la mayoría de las exploraciones en el mundo de linux, estará interactuando con el sistema a través del uso de un intérprete de comandos. Un intérprete de comandos es simplemente un programa que toma la entrada del usuario (p.ej. las órdenes que teclea) y las traduce a instrucciones. Esto puede ser comparado con el COMMAND.COM de MS-DOS, el cual efectúa esencialmente las mismas tareas. El intérprete de comandos es sólo uno de las interfaces con linux. Hay muchas interfaces posibles como el sistema X Windows, el cual le permite ejecutar comandos usando el ratón y el teclado.

Tan pronto entra en el sistema, se arranca un intérprete de comandos desde donde puede teclear órdenes al sistema.

Vea el siguiente ejemplo:

```
telsur login:   admin
Password:
Welcome to telsur!

/home/admin#
```

Aquí, admin entra en el sistema y es situado en el intérprete de comandos, "/home/admin#" es el "prompt" del intérprete de comandos, indicando que esta listo para recibir órdenes. Trate de decirle al sistema que haga algo interesante:

```
/home/admin# make cine
make:   *** No se puede ir al cine.   Stop.
/home/admin#
```

Como resulta que make es el nombre de un programa ya existente en el sistema, el intérprete de comandos lo ejecuta.

Esto nos lleva a una cuestión importante: >Que son órdenes? >Que ocurre cuando tecleamos "make cine"?, la primera palabra de la orden, "make", es el nombre de la orden a ejecutar, el resto de la orden es tomado como argumentos de la propia orden. Ejemplos:

```
/home/admin# cp tesis
```

Aquí, el nombre de la orden es "cp", y el argumento “tesis”. Cuando tecleamos una orden, el intérprete de comandos hace varias cosas. Primero que nada, busca el nombre de la orden y comprueba si es una orden interna. (Es decir, una orden que el propio intérprete de comandos sabe ejecutar por si mismo. Hay bastantes órdenes de ese tipo que verá más adelante). El intérprete de comandos también comprueba si la orden es un "alias" o nombre sustituto de otra orden. Si no se cumple ninguno de estos casos, el intérprete de comandos busca el programa y lo ejecuta pasándole los argumentos especificados en la línea de comandos.

En el ejemplo anterior, el intérprete de comandos busca el programa llamado make y lo ejecuta con el argumento cine, make es un programa usado a menudo para compilar programas grandes, y toma como argumentos el nombre de un "objetivo" a compilar. En el caso de "make cine", le está ordenando a make que compile el objetivo cine. Como make no puede encontrar un objetivo de ese nombre, falla enviando un mensaje de error y volviendo al intérprete de comandos. Ahora, que ocurre si se da una orden y el intérprete de comandos no puede encontrar el programa de ese nombre, bastante simple, si no se puede encontrar el programa con el nombre dado en la orden, se muestra un mensaje de error que debería ser auto explicativo. A menudo verá este mensaje de error si se equivoca al teclear una orden (por ejemplo, si hubiese tecleado "mkae cine" en lugar de "make cine".

8.4 Salida del sistema

Antes de ahondar más, debe ver como salir del sistema, desde la línea de órdenes use la orden para salir. Hay otras formas, pero esta es la más fácil.

```
/home/admin# exit
```

8.5 Cambiando la palabra de paso

También debe asegurarse de saber como cambiar su palabra de paso. La orden passwd le pedirá su palabra de paso vieja y la nueva, volverá a pedir una segunda vez la nueva para validarla. Es necesario no olvidar la palabra de paso si eso ocurre, puede cambiarla como administrador del sistema, tema que se verá más adelante.

9. Ficheros y directorios

Bajo la mayoría de los sistemas operativos (linux incluido), existe el concepto de fichero, el cual es un conjunto de información al que se le ha asignado un nombre (llamado nombre del fichero).

Ejemplos de fichero son un mensaje de correo, o un programa que puede ser ejecutado, esencialmente, cualquier cosa salvada en el disco es guardada en un fichero individual.

Los ficheros son identificados por sus nombres, por ejemplo, el fichero que contiene algún historial podría ser salvado con el nombre history. Estos nombres usualmente identifican el fichero y su contenido de alguna forma significativa para quien los crea. No hay un formato estándar para los nombres de los ficheros como lo hay en MS-DOS y en otros sistemas operativos; en general estos pueden contener cualquier carácter (excepto /), y están limitados a 256 caracteres de longitud.

Con el concepto de fichero aparece el concepto de directorio. Un directorio es simplemente una colección de ficheros. Puede ser considerado como una "carpeta" que contiene muchos ficheros diferentes. Los directorios también tienen nombre con el que los que

se pueden identificar. Además, los directorios mantienen una estructura de árbol; es decir, los directorios pueden contener otros directorios.

Un fichero puede ser referenciado por su nombre con camino, el cual esta constituido por su nombre, antecedido por el nombre del directorio que lo contiene. Por ejemplo, suponga que admin tiene un directorio de nombre “papers” que contiene tres ficheros: “atm-final”, “sdh-lit” y “masters-tesis”. (Cada uno de los tres ficheros contiene información sobre tres de los proyectos en los que admin esta trabajando). Para referirse al fichero sdh-lit, admin puede especificar su camino:

papers/sdh-lit

Como se puede ver, el directorio y el nombre del fichero van separados por un carácter /. Por esta razón, los nombres de fichero no pueden contener este carácter. Los usuarios de MS-DOS encontraran esta convención familiar, aunque en el mundo MS-DOS se usa el carácter \).

Como se ha mencionado, los directorios pueden anidarse uno dentro de otro, por ejemplo, suponga que admin tiene otro directorio dentro de papers llamado cheat-sheet. El camino de este fichero sería:

papers/notes/cheat-sheet

Por lo tanto, el camino realmente es la "ruta" que se debe tomar para localizar a un fichero. El directorio sobre un subdirectorio dado es conocido como el directorio padre. Aquí, el directorio papers es el padre del directorio notes.

9.1 El árbol de directorios

La mayoría de los sistemas linux tienen una distribución de ficheros estándar, de forma que recursos y ficheros puedan ser fácilmente localizados. Esta distribución forma el árbol de directorios, el cual comienza en el directorio "/", también conocido como "directorio raíz". Directamente por debajo de / hay algunos subdirectorios importantes: /bin, /etc, /dev y /usr, entre otros. Estos a su vez contienen otros directorios con ficheros de configuración del sistema, programas, etc.

En particular, cada usuario tiene un directorio "home". Este es el directorio en el que el usuario guardará sus ficheros. En los ejemplos anteriores, todos los ficheros de admin (como cheat-sheer y sdh-lit) estaban contenidos en el directorio home de admin. Usualmente, los directorios home de los usuarios cuelgan de /home y son nombrados con el nombre del usuario al que pertenecen. Por lo tanto, el directorio "home" de admin es /home/admin.

En la figura 2 se muestra un árbol de directorio de ejemplo. Este debería dar una idea de como está organizado en el sistema el árbol de directorios.

9.2 Directorio de trabajo actual

En cualquier momento, las órdenes que se teclean al intérprete de comandos son dadas en términos de su directorio de trabajo actual. Puede pensar en el directorio actual de trabajo como en el directorio en el que actualmente se está "situado". Cuando entra en el sistema, el directorio de trabajo se inicia en el directorio home/admin (en este caso). En cualquier momento que se refiera a un fichero, puede hacerlo en relación al directorio de trabajo actual, en lugar de especificar el camino completo del fichero. Vemos un ejemplo. Admin tiene el directorio papers, y papers contiene el fichero sdh-lit, si admin quiere echar un vistazo a ese fichero, puede usar la orden

```
/home/admin# more /home/admin/papers/sdh-lit
```

La orden more simplemente muestra el fichero, pantalla a pantalla, pero, como el directorio de trabajo actual de admin es /home/admin, podría haberse referido al fichero de forma relativa a su directorio de trabajo actual. La orden seria

```
/home/admin# more papers/sdh-lit
```

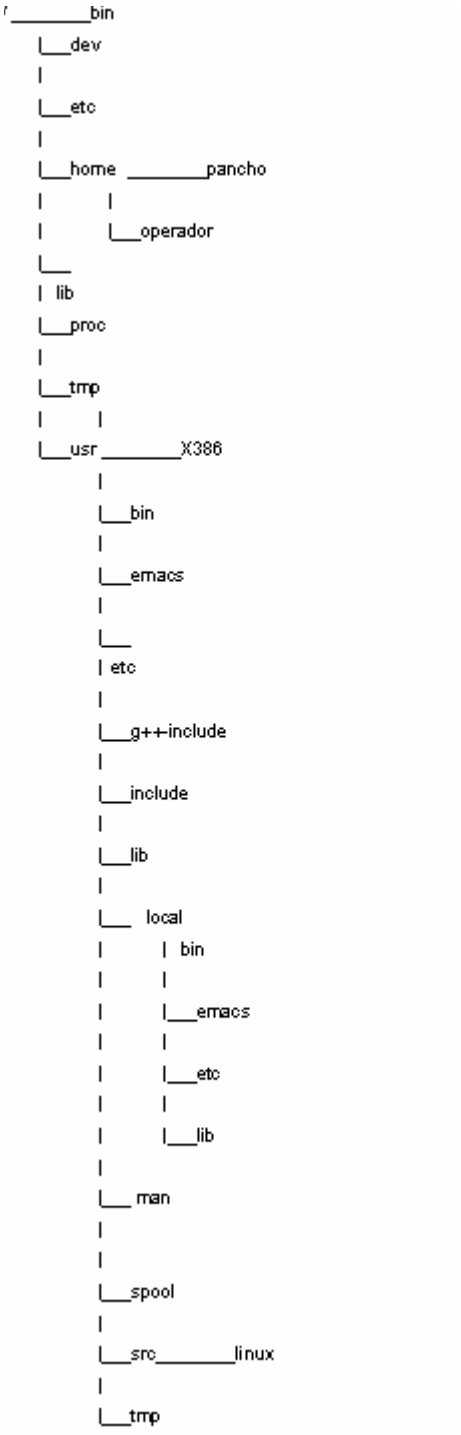


Figura 2: Típico árbol de directorios Linux (resumido).

Por lo tanto, si se comienza el nombre de un fichero (como papers/final) con un carácter distinto a "/", el sistema supone que se está refiriendo al fichero con su posición relativa a su directorio de trabajo. Esto es conocido como camino relativo.

Por otra parte, si comienza el nombre del fichero con "/", el sistema interpreta esto como un camino completo es decir, el camino al fichero completo desde el directorio raíz, /. Esto es conocido como camino absoluto.

9.3 Refiriéndose al directorio home

Bajo tcsh y bash,[7] el directorio "home" puede ser referenciado usando el carácter de la tilde (~). Por ejemplo, la orden

```
/home/admin# more "/papers/sdh-lit
```

es equivalente a

```
/home/admin# more /home/admin/papers/sdh-lit
```

El carácter "~" es simplemente sustituido por el intérprete de comandos, con el nombre del directorio home. Además, también puede especificar otros directorios home de usuarios con la tilde. El camino "~francisco/letters" es traducido por el intérprete de órdenes a "/home/francisco/letters" (si /home/francisco es el directorio home de Francisco). El uso de la tilde es simplemente un atajo; no existe ningún directorio llamado "~" es simplemente una ayuda sintáctica proporcionada por el intérprete de comandos.

[7] tcsh y bash son dos intérpretes de comandos que corren bajo Linux. Un intérprete de comandos es el programa que lee las órdenes del usuario y las ejecuta; la mayoría de los sistemas Linux habilitan tcsh o bash para las nuevas cuentas de usuario.

10. Primeros pasos en Linux

Antes de comenzar es importante destacar que todos los nombres de ficheros y comandos son "case-sensitive" (que hace diferencia entre mayúsculas y minúsculas, a diferencia de sistemas operativos como MS-DOS). Por ejemplo, el comando make es diferente a Make o MAKE. Lo mismo ocurre en el caso de nombres de ficheros o directorios.

10.1 Moviéndonos por el entorno

Ahora que ya puede presentarse como usuario, y sabe como indicar ficheros con su camino completo, ¿Como puede cambiar su directorio de trabajo?

La orden para moverse por la estructura de directorios es cd, abreviación de "cambio de directorio". Hay que destacar, que la mayoría de las ordenes linux más usadas son de dos o tres letras. La forma de uso de la orden cd es:

cd <directorio>

donde <directorio> es el nombre del directorio al que se desea ir.
Como se dijo, al entrar al sistema se comienza en el directorio "home". Si Admin quiere ir al subdirectorio papers, debería usar la orden

```
/home/admin# cd papers
/home/admin/papers#
```

Como se puede ver, la línea de comandos de Admin cambia para mostrar el directorio actual de trabajo. Ahora que ya está en el directorio papers puede echarle un vistazo al fichero sdh-lit con el comando *more*

```
/home/admin/papers# more sdh-lit
```

Ahora, Admin está en el subdirectorio papers, para volver al directorio padre de éste, se usa la orden:

```
/home/admin/papers# cd ..
/home/admin#
```

(Es importante el espacio entre "cd" y ".."). Cada directorio tiene una entrada de nombre ".." la cual se refiere al directorio padre. De igual forma, existe en cada directorio la entrada "." la cual se refiere a si mismo. Así que el comando lo deja donde estaba.

```
/home/admin/papers# cd .
/home/admin#
```

También pueden usarse nombres con el camino absoluto en la orden cd. Para ir al directorio de Francisco con cd, se introduce la siguiente orden.

```
/home/admin/papers# cd /home/francisco
/home/francisco#
```

También, usando cd sin argumentos lo llevará al directorio de origen.

```
/home/francisco# cd
/home/admin#
```

10.2 Mirando el contenido de los directorios

El simple movimiento por el árbol de directorios es poco útil, necesita un nuevo comando, ls. ls muestra por el terminal la lista de ficheros y directorios, por defecto, los del directorio activo. Por ejemplo;

```
/home/admin# ls
Mail
```

```
letters
papers
/home/admin#
```

Aquí puede ver que Admin tiene tres entradas en su directorio actual: Mail, letters y papers.

Esto no dice demasiado ¿Son ficheros o directorios?. Puede usar la opción -F de la orden ls para obtener más información.

```
/home/admin# ls -F
Mail/
letters/
papers/
/home/admin#
```

Por el carácter / añadido a cada nombre se sabe que las tres entradas son subdirectorios.

La orden ls -F puede también añadir al final "*", esto indica que es un fichero ejecutable. Si ls -F no añade nada, entonces es un fichero normal, es decir, no es ni un directorio ni un ejecutable.

Por lo general, cada orden Linux puede tomar una serie de opciones definidas en forma de argumentos. Estos usualmente comienzan con el carácter "-", como se vio anteriormente con ls -F. La opción -F le dice a ls que de más información sobre el tipo de ficheros, en este caso añadiendo un / detrás de cada nombre de un directorio.

Si a ls le pasa un nombre de directorio, mostrará el contenido de ese directorio.

```
/home/admin# ls -F papers
Sdh-lit
tesis-final
masters-thesis
notes/
/home/admin#
```

Para ver un listado más completo, vea el contenido del directorio del sistema /etc.

```
/home/admin# ls /etc

Images      ftpusers    lpc         rc.new      shells
adm         getty       magic       rc0.d       startcons
bcheckrc    gettydefs   motd        rc1.d       swapoff
brc         group       mount       rc2.d       swapon
brc~        inet        mtab        rc3.d       syslog.conf
csh.cshrc   init        mtools      rc4.d       syslog.pid
csh.login   init.d      pac         rc5.d       syslogd.reload
default     initrunlvl  passwd      rmt         termcap
disktab     inittab     printcap    rpc         umount
fdprm       inittab.old profile     rpcinfo     update
fstab       issue       psdatabase  securetty   utmp
ftpaccess   lilo        rc          services    wtmp

/home/admin#
```

(Para los usuarios de MS-DOS, nótese que los nombres de los ficheros pueden ser mayores de caracteres y pueden contener puntos en cualquier posición. Incluso es posible que un fichero contenga más de un punto en su nombre.)
Puede llegar al directorio raíz con "cd .." y desde allí ir al directorio /usr/bin.

```
/home/admin# cd ..
/home# cd ..
/# cd usr
/usr# cd bin
/usr/bin#
```

También puede moverse dentro de los directorios en múltiples pasos, como en cd /usr/bin. Si se trata de mover por varios directorios usando ls y cd. En algunos casos podrá encontrarse con el mensaje de error "Permission denied". Esto simplemente es debido a cuestiones de seguridad del linux. Para poder moverse o listar un directorio debe tener permisos para poder hacerlo. Se hablará sobre ello más adelante.

10.3 Creando directorios nuevos

Es el momento de aprender a crear directorios. Para ello se usa la orden mkdir:

```
/home/admin# mkdir foo
/home/admin# ls -F
Mail/
foo/
letters/
papers/
/home/admin# cd foo
/home/admin/foo# ls
/home/admin/foo#
```

Acaba de crear un directorio nuevo y entró a el. Como no hay ningún fichero en el directorio nuevo, verá como copiar ficheros desde un lugar a otro.

10.4 Copia de ficheros

La copia de ficheros es efectuada por la orden cp:

```
/home/admin/foo# cp /etc/termcap  .
/home/admin/foo# cp /etc/shells  .
/home/admin/foo# ls -F
shells      termcap
```

```
/home/admin/foo# cp shells bells
/home/admin/foo# ls -F
bells     shells     termcap
/home/admin/foo#
```

La orden cp copia los ficheros listados en la línea de comandos al fichero o directorio pasado como ultimo argumento. Nótese como se usa el directorio "." para referirse al directorio actual.

10.5 Moviendo ficheros

La orden mv mueve ficheros en lugar de copiarlos. La sintaxis es muy sencilla.

```
/home/admin/foo# mv termcap sells
/home/admin/foo# ls -F
bells     sells     shells
/home/admin/foo#
```

Nótese como termcap ya no existe, en su lugar está el fichero sells. Esta orden puede usarse para renombrar ficheros, como acaba de hacer, pero también para mover ficheros a directorios diferentes.

Nota: mv y cp sobrescribirán los ficheros destino (si ya existen) sin consultar. Se debe tener cuidado cuando se mueven los ficheros a otro directorio: puede haber ya un fichero con el mismo nombre que será sobrescrito.

10.6 Borrando ficheros y directorios

Para borrar un fichero, use la orden rm. ("rm" viene de "remove").

```
/home/admin/foo# rm bells sells
```

```
/home/admin/foo# ls -F
shells
/home/admin/foo#
```

Se ha quedado sólo con el fichero "shells". Nótese que rm por defecto no preguntará antes de borrar un fichero luego, sea cuidadoso.

Una orden relacionada con rm es rmdir. Esta orden borra un directorio, pero sólo si está vacío. Si el directorio contiene ficheros o subdirectorios, rmdir reclamará.

10.7 Mirando los ficheros

Las órdenes more y cat son usadas para ver el contenido de ficheros. more muestra el fichero pantalla a pantalla mientras que cat lo muestra entero de una vez.
Para ver el contenido del fichero shells podemos usar la orden

```
/home/admin/foo# more shells
```

El contenido de shells, es una lista de intérpretes de comandos validos disponibles en el sistema. En la mayoría de los sistemas incluye /bin/sh, /bin/bash y /bin/csh.
Se hablará sobre los diferentes intérpretes de comandos más adelante.

Durante la ejecución de more se pulsa la barra de espacios para avanzar a la página siguiente y *b* para volver a la página anterior. Hay otros comandos disponibles, los citados son solo los más básicos *q* finalizará la ejecución de more.

Con cat, el texto probablemente pasará demasiado rápido como para poder leerlo. El nombre "cat" viene de "concaténate", que es para lo que realmente sirve el programa. La orden cat puede ser usada para concatenar el contenido de varios ficheros y guardar el resultado en otro fichero. Esto se discutirá más adelante.

10.8 Obteniendo ayuda en línea

Prácticamente cada sistema UNIX, incluido Linux, proporciona una utilidad conocida como "páginas de manual". Estas páginas contienen documentación en línea para todas las órdenes del sistema, recursos, ficheros de configuración, etc.

La orden usada para acceder a las páginas de manual es man. Por ejemplo, si está interesado en conocer otras opciones de la orden ls, puede escribir

```
/home/admin# man ls
```

y le mostrará la página del manual para el comando ls.

Desafortunadamente la mayoría de las páginas de manual han sido escritas por personas que ya conocían lo que la orden o recurso hacían, por esto, las páginas de manual usualmente sólo contienen detalles técnicos de la orden sin ningún tipo de tutorial de uso. Pese a esto, estas páginas son una gran fuente de información que permite refrescar la memoria si olvida la sintaxis de un comando. Igualmente, estas páginas darán mucha información sobre órdenes que no se tratarán en esta tesis.

11. Sumario de Órdenes Básicas

Esta sección introduce algunos de las órdenes básicas más útiles de un sistema linux, incluidas las ya cubiertas en las secciones anteriores.

Nótese que las opciones usualmente comienzan con "-" y en la mayoría de los casos se pueden añadir múltiples opciones de una letra con un único "-". Por ejemplo, en lugar de usar ls -l -F es posible usar ls -lF.

En lugar de listar todas las opciones disponibles para cada uno de los comandos sólo se verán aquellas más útiles o importantes. De hecho, la mayoría de las ordenes tienen un gran número de opciones (muchas de las cuales nunca se usan). Puede usar man para ver las páginas de manual de cada orden, la cual mostrará la lista completa de opciones disponibles.

Nótese también, que la mayoría de las órdenes toman una lista de ficheros o directorios como argumentos, denotados como "<fichero1> . . . <ficheroN>". Por ejemplo, la orden cp toma

como argumentos la lista de ficheros a copiar, seguidos del fichero o directorio destino. Cuando se copia más de un fichero, el destino debe de ser un directorio.

cd: Cambia el directorio de trabajo actual.
Sintaxis: cd <directorio>
<directorio> es el directorio al que cambia. ("." se refiere al directorio actual,
".." al directorio padre.)
Ejemplo: cd ../foo pone ../foo como directorio actual.

ls: Muestra información sobre los ficheros o directorios indicados.
Sintaxis: ls <fichero1> <fichero2> ...<ficheroN>
Donde <fichero1> a <ficheroN> son los ficheros o directorios a listar.
Opciones: Hay mas opciones de las que podría suponer. Las más usadas comúnmente son: -F (usada para mostrar información sobre el tipo de fichero), y -l (da un listado "largo" incluyendo tamaño, propietario, permisos, etc. Se tratará esto en detalle mas adelante.)
Ejemplo: ls -lF /home/admin mostrará el contenido del directorio /home/admin.

cp: Copia fichero(s) en otro fichero o directorio.
Sintaxis: cp <fichero1> <fichero2> ...<ficheroN> <destino>
Donde <fichero1> a <ficheroN> son los ficheros a copiar, y <destino> es el fichero o directorio destino.
Ejemplo: cp ../frog cnt copia el fichero ../frog al fichero o directorio cnt.

mv: Mueve fichero(s) a otro fichero o directorio. Es equivalente a una copia seguida del borrado del original. Puede ser usado para renombrar ficheros, como el comando MS-DOS RENAME.
Sintaxis: mv <fichero1> <fichero2> ...<ficheroN> <destino>
Donde <fichero1> a <ficheroN> son los ficheros a "mover" y <destination> es el fichero o directorio destino.
Ejemplo: mv ../frog cnt mueve el fichero ../frog al fichero o directorio cnt.

rm: Borra ficheros. Nótese que cuando los ficheros son borrados en linux, son irrecuperables (a diferencia de MS-DOS, donde usualmente se puede recuperar un fichero borrado).

Sintaxis: rm <fichero1> <fichero2> ...<ficheroN>
Donde <fichero1> a <ficheroN> son los nombres de los ficheros a borrar.
Opciones: -i pedirá confirmación antes de borrar un fichero. Ejemplo: rm -i /home/admin/cnt /home/admin/frog borra los ficheros cnt y frog en /home/admin.

mkdir: Crea directorios nuevos.
Sintaxis: mkdir <dir1> <dir2> ...<dirN>
Donde <dir1> a <dirN> son los directorios a crear.
Ejemplo: mkdir /home/admin/test crea el directorio test colgando de /home/admin.

rmdir: Esta orden borra directorios vacíos. Al usar rmdir, el directorio de trabajo actual no debe de estar dentro del directorio a borrar.
Sintaxis: rmdir <dir1> <dir2> ...<dirN>
Donde <dir1> a <dirN> son los directorios a borrar.
Ejemplo: rmdir /home/admin/papers borra el directorio /home/admin/papers si esta vacío.

man: Muestra la página de manual del comando o recurso (cualquier utilidad del sistema que no es un comando, como funciones de librería) dado. Sintaxis: man <command>
Donde <command> es el nombre del comando o recurso sobre el que quiere obtener la ayuda
Ejemplo: man ls muestra ayuda sobre la orden ls.

more: Muestra el contenido de los ficheros indicados, una pantalla cada vez.
Sintaxis: more <fichero1> <fichero2> ...<ficheroN>
Donde <fichero1> a <ficheroN> son los ficheros a mostrar.
Ejemplo: more papers/sdh-lit muestra por el terminal el contenido del fichero papers/sdh-lit.

cat: Oficialmente usado para concatenar ficheros, cat también es usado para mostrar el contenido completo de un fichero de una vez.
Sintaxis: cat <fichero1> <fichero2> ...<ficheroN>
Donde <fichero1> a <ficheroN> son los ficheros a mostrar.
Ejemplo: cat letters/from-mdw muestra por el terminal el contenido del fichero letters/from-mdw.

echo: Simplemente envía al terminal los argumentos pasados.

Sintaxis: echo <arg1> <arg2> ...<argN>

Donde <arg1> a <argN> son los argumentos a mostrar.

Ejemplo: echo ``Hola mundo'' muestra la cadena "Hola mundo".

grep: Muestra todas las líneas de un fichero dado que coinciden con un cierto patrón.

Sintaxis: grep <patrón> <fichero1> <fichero2> ...<ficheroN>

Donde <patrón> es una expresión regular y <fichero1> a <ficheroN> son los ficheros donde buscar.

Ejemplo: grep loomer /etc/hosts mostrará todas las líneas en el fichero /etc/hosts que contienen la cadena "loomer".

CAPITULO IV
TRABAJANDO CON FICHEROS

12. Explorando el Sistema de Ficheros

El sistema de ficheros es la colección de ficheros y la jerarquía de directorios del sistema.

Primero, deberá cambiarse al directorio raíz (cd /) y ejecutar ls -F. Probablemente verá estos directorios[8]:

bin, dev, etc, home, install, lib, mnt, proc, root, tmp, user, usr, y var.

A continuación se verá el detalle de cada uno de estos directorios.

Tabla 5: Árbol de Directorios

/bin	/bin es la abreviación de "binaries", o ejecutables. Es donde residen la mayoría de los programas esenciales del sistema. Use la orden "ls -F /bin" para listar los ficheros. Puede ver algunas ordenes que reconocerá, como cp, ls y mv. Estos son los programas para estas órdenes. Cuando usa la orden cp esta ejecutando el programa /bin/cp. Usando ls -F verá que la mayoría (si no todos) los ficheros de /bin tienen un asterisco ("*") añadido al final de sus nombres. Esto indica que son ficheros ejecutables, como se describirá más adelante.
/dev	Los "ficheros" en /dev son conocidos como controladores de dispositivo (device drivers) son usados para acceder a los dispositivos del sistema y recursos, como discos duros, modems, memoria, etc. Por ejemplo, de la misma forma que se puede leer

[8] Puede ver otros o incluso no ver todos. Cada versión de Linux difiere en algunos aspectos.

	datos de un fichero, puede leerla desde la entrada del ratón leyendo /dev/mouse. Los ficheros que comienzan su nombre con fd son controladores de disqueteras. fd0 es la primera disquetera, fd1 la segunda. Ahora, puede darse cuenta de que hay más controladores de dispositivo para disqueteras de los que se han mencionado. Estos representan tipos específicos de discos. Por ejemplo, fd1H1440 accederá a discos de 3.5" de alta densidad en la disquetera 1. Aquí tiene una lista de algunos de los controladores de dispositivo más usados. Nótese que incluso aunque puede que no se tenga alguno de los dispositivos listados, aparecen entradas en dev de cualquier forma. /dev/console hace referencia a la consola del sistema, es decir, al monitor conectado directamente a su sistema. Los dispositivos /dev/ttyS y /dev/cua son usados para acceder a los puertos serie. Por ejemplo, /dev/ttyS0 hace referencia a "COM1" bajo MS-DOS. Los dispositivos /dev/cua son "callout", los cuales son usados en conjunción con un módem. Los nombres de dispositivo que comienzan por hd acceden a discos duros. /dev/hda hace referencia a la totalidad del primer disco duro, mientras que /dev/hda1 hace referencia a la primera partición en /dev/hda. Los nombres de dispositivo que comienzan con sd son dispositivos SCSI. Si tiene un disco duro SCSI, en lugar de acceder a él mediante /dev/hda, debe acceder a /dev/sda. Las cintas SCSI son accedidas vía dispositivos st y los CD-ROM SCSI vía sr. Los nombres que comienzan por lp acceden a los puertos paralelos. /dev/lp0 hace referencia a "LPT1" en el mundo MS-DOS. /dev/null es usado como "agujero negro" cualquier dato enviado a este dispositivo desaparece. ¿Para que puede ser útil esto?. Bien, si desea suprimir la salida por pantalla de una orden, podría enviar la salida a /dev/null. Se analizará más sobre esto después. Los nombres que comienzan por /dev/tty hacen referencia a "consolas virtuales" de su sistema (accesibles mediante las teclas

	alt-F1, *alt-F2,* etc). /dev/tty1 hace referencia a la primera VC, /dev/tty2 a la segunda, etc. Los nombres de dispositivo que comienzan con /dev/pty son "pseudo-terminales". Estos son usados para proporcionar un "terminal" a sesiones remotas. Por ejemplo, si la máquina está en una red, el telnet de entrada usará uno de los dispositivos /dev/pty.
/etc	/etc contiene una serie de ficheros de configuración del sistema. Estos incluyen /etc/passwd (la base de datos de usuarios), /etc/rc (guiones de inicialización del sistema), etc.
/sbin	/sbin se usa para almacenar programas esenciales del sistema, que usará el administrador del sistema.
/home	/home contiene los directorios "home" de los usuarios. Por ejemplo, /home/admin es el directorio del usuario "admin". En un sistema recién instalado, no habrá ningún usuario en este directorio.
/lib	/lib contiene las imágenes de las librerías compartidas. Estos ficheros contienen códigos que compartirán muchos programas. En lugar de que cada programa contenga una copia propia de las rutinas compartidas, éstas son guardadas en un lugar común, en /lib. Esto hace que los programas ejecutables sean menores y ocupan un lugar común, en /lib.
/proc	/proc es un "sistema de ficheros virtual". Los ficheros que contiene realmente residen en memoria, no en un disco. Hacen referencia a varios procesos que corren en el sistema, y le permiten obtener información acerca de que programas y procesos están corriendo en un momento dado.
/tmp	Muchos programas tienen la necesidad de generar cierta información temporal y guardarla en un fichero temporal. El lugar habitual para esos ficheros es en /tmp.
/usr	/usr es un directorio muy importante. Contienen una serie de subdirectorios que contienen a su vez algunos de los más importantes y útiles programas y ficheros de configuración usados en el sistema. Los directorios descritos anteriormente son

	esenciales para que el sistema este operativo, pero la mayoría de las cosas que se encuentran en /usr son opcionales para el sistema. De cualquier forma, son estas cosas opcionales las que hacen que el sistema sea útil e interesante. Sin /usr, tendría un sistema aburrido, solo con programas como cp y ls. /usr contiene la mayoría de los paquetes grandes de programas y sus ficheros de configuración.
/usr/X386	/usr/X386 contiene el sistema X Window si decide instalarlo. El sistema X Window es un entorno gráfico grande y potente el cual proporciona un gran número de utilidades y programas gráficos, mostrados en "ventanas" en la pantalla. Si está familiarizado con los entornos Microsoft Windows o Macintosh, X Window le será muy familiar. El directorio /usr/X386 contiene todos los ejecutables de X Window, ficheros de configuración y de soporte.
/usr/bin	/usr/bin es el almacén real de programas del sistema linux. Contiene la mayoría de los programas que no se encuentran en otras partes como /bin.
/usr/etc	Como /etc contiene diferentes ficheros de configuración y programas del sistema, /usr/etc contiene incluso más que el anterior. En general, los ficheros que se encuentran en /usr/etc/ no son esenciales para el sistema, a diferencia de los que se encuentran en /etc, que si lo son.
/usr/include	/usr/include contiene los ficheros de cabecera para el compilador de C. Estos ficheros (la mayoría de los cuales terminan en .h, de "header") declaran estructuras de datos, sub-rutinas y constantes usados en la escritura de programas en C. Los ficheros que se encuentran en /usr/include/sys son generalmente usados en la programación de linux a nivel de sistema. Si se está familiarizado con el lenguaje de programación C, aquí encontrará los ficheros de cabecera como stdio.h, el cual declara funciones como printf().
/usr/g++-include	/usr/g++-include contiene ficheros de cabecera para el compilador de C++ (muy parecido a /usr/include).

/usr/lib	/usr/lib contiene las librerías equivalentes "stub" y "static" a los ficheros encontrados en /lib. Al compilar un programa, este es "enlazado" con las librerías que se encuentran en /usr/lib, las cuales dirigen al programa a buscar en /lib cuando necesita el código de la librería. Además, varios programas guardan ficheros de configuración en /usr/lib.
/usr/local	/usr/local es muy parecido a /usr contiene programas y ficheros no esenciales para el sistema, pero que hacen el sistema mas robusto. En general, los programas que se encuentran en /usr/local son específicos del sistema, esto es, el directorio /usr/local que difiere bastante entre sistemas UNIX. Aquí encuentra programas grandes como TEX (sistema de formateo de documentos) y Emacs (gran y potente editor), si los instala.
/usr/man	Este directorio contiene las páginas de manual. Hay dos subdirectorios para cada página "sección" de las páginas (ejecutando la orden man para ver mas detalles). Por ejemplo, /usr/man/man1 contiene los fuentes (es decir, los originales por formatear) de las páginas de manual de la sección 1, y /usr/man/cat1 las páginas ya formateadas de la sección 1.
/usr/src	/usr/src contiene el código fuente (programas por compilar) de varios programas de su sistema. El más importante es /usr/src/Linux, el cual contiene el código fuente del núcleo de Linux.
/var	/var contiene directorios que a menudo cambian su tamaño o tienden a crecer. Muchos de estos directorios solían residir en /usr, lo que se está tratando de dejar relativamente inalterable, los directorios que cambian a menudo han sido llevados a /var. Algunos de estos directorios son: /var/adm /var/adm contiene varios ficheros de interés para el administrador del sistema, específicamente históricos, los cuales recogen errores o problemas con el sistema. Otros ficheros guardan las sesiones de presentación en el sistema, así como los intentos fallidos. /var/spool contiene ficheros que van a ser pasados a otro programa. Por ejemplo, si la máquina está conectada a una red, el

	correo de llegada será almacenado en /var/spool/Mail hasta que lo lea o borre. Artículos nuevos de las news" tanto salientes como entrantes pueden encontrarse en /var/spool/news, etc.

13. Tipos de intérpretes de comandos

Como se ha mencionado anteriormente y en numerosas ocasiones, UNIX es un sistema operativo multitarea y multiusuario. La multitarea es muy útil, en poco tiempo se pueden ejecutar programas "de fondo", conmutar entre múltiples tareas y "entubar" programas unos entre otros para conseguir resultados complejos con un único comando.

Muchas de las características que se tratará en esta sección son proporcionadas por el intérprete de comandos. Hay que tener cuidado en no confundir Linux (el sistema operativo) con el intérprete de comandos, éste ultimo, es un interfase con el sistema que hay debajo. El intérprete de comandos proporciona la funcionalidad sobre Linux, este, no es sólo un intérprete interactivo de los comandos, sino también un potente lenguaje de programación, mediante el cual puede escribir guiones, que permiten juntar varias órdenes en un fichero. Los usuarios de MS-DOS reconocerán esto como los ficheros "batch". El uso de los guiones del intérprete de comandos es una herramienta muy poderosa que le permitirá automatizar e incrementar el uso de Linux.

Hay varios tipos de intérpretes de comandos en el mundo linux. Los dos más importantes son el "Bourne shell" y el "C shell". El intérprete de comandos Bourne, usa una sintaxis de comandos como la usada en los primeros sistemas UNIX, el System III. El nombre del interprete Bourne en la mayoría de los Linux es /bin/sh (donde sh viene de "shell", intérprete de comandos en inglés). El interprete C usa una sintaxis diferente, a veces parecida a la del lenguaje de programación C, y en la mayoría de los sistemas UNIX se encuentra como /bin/csh.

Bajo Linux hay algunas diferencias en los intérpretes de comandos disponibles. Dos de los más usados son el "Bourne Again Shell" o "Bash" (/bin/bash) y Tcsh (/bin/tcsh). Bash es un equivalente al Bourne con muchas características avanzadas de la C shell. Como Bash es un súper-conjunto de la sintaxis del Bourne, cualquier guión escrito para el intérprete de comandos Bourne standard funcionará en Bash. Para los que prefieren el uso del intérprete de comandos C, Linux tiene el Tcsh, que es una versión extendida del C original.

El tipo de intérprete de comandos que decida usar es puramente una cuestión de gustos. Algunas personas prefieren la sintaxis del Bourne con las características avanzadas que proporciona Bash, y otros prefieren el más estructurado intérprete de comandos C. En lo que respecta a los comandos usuales como cp, ls, etc, es indiferente el tipo de intérprete de comandos usado, la sintaxis es la misma. Sólo, cuando se escriben guiones para el intérprete de comandos, o se usan características avanzadas aparecen las diferencias entre los distintos intérpretes de comandos.

13.1 Caracteres comodín

Una característica importante de la mayoría de los intérpretes de comandos en UNIX es la capacidad para referirse a más de un fichero usando caracteres especiales. Estos llamados comodines permiten referirse, por ejemplo, a todos los ficheros que contienen el carácter “n".

El comodín "*" hace referencia a cualquier carácter o cadena de caracteres en el fichero, por ejemplo, cuando se usa el carácter "*" en el nombre de un fichero, el intérprete de comandos lo sustituye por todas las combinaciones posibles provenientes de los ficheros en el directorio al cual se está refiriendo.

Por ejemplo. Suponga que Admin tiene los ficheros frog, cnt y stuff en el directorio actual.

```
/home/admin# ls
frog    cnt    stuff
/home/admin#
```

Para acceder a todos los ficheros con la letra "o" en su nombre, ha de usar la orden

```
/home/admin# ls *o*
frog
/home/admin#
```

Como se puede ver, el comodín "*" ha sido sustituido con todas las combinaciones posibles que coincidían de entre los ficheros del directorio actual. El uso de "*" solo, simplemente se refiere a todos los ficheros, puesto que todos los caracteres coinciden con el comodín.

```
/home/admin# ls *
frog      cnt      stuff
/home/admin#
```

Más ejemplos:

```
/home/admin# ls f*
frog
/home/admin# ls *ff
stuff
/home/admin# ls *f*
frog      stuff
/home/admin# ls s*f
stuff
/home/admin#
```

El proceso de la sustitución de "*" en nombres de ficheros es llamado expansión de comodines y es efectuado por el intérprete de comandos. Esto es importante: las órdenes individuales, como ls, nunca ven el "*" en su lista de parámetros. Es el intérprete quien expande los comodines para incluir todos los nombres de ficheros que se adaptan. Luego la orden:

```
/home/admin# ls *o*
```

es expandida para obtener

```
/home/admin# ls frog
```

Una nota importante acerca del carácter comodín "*". El uso de este comodín NO cuadrará con nombres de ficheros que comiencen con un punto ("."). Estos ficheros son

tratados como "ocultos" aunque no lo están realmente, simplemente no son mostrados en un listado normal de ls y no son afectados por el uso del comodín "*".

He aquí un ejemplo. Ya se ha mencionado que cada directorio tiene dos entradas especiales: ".", que hace referencia al directorio actual y "..", que se refiere al directorio padre. De cualquier forma, cuando usa “ls” estas dos entradas no se muestran.

```
/home/admin# ls
frog    cnt    stuff
/home/admin#
```

Si usa el parámetro -a con ls podrá ver nombres de ficheros que comienzan por ".".
Ejemplo:

```
/home/admin# ls -a
.    ..    .bash_profile    .bashrc    frog    cnt    stuff
/home/admin#
```

Ahora podrá ver las dos entradas especiales, "." y "..", así como otros dos ficheros "ocultos" .bash_profile y .bashrc. Estos dos ficheros son usados en el arranque por bash cuando admin se presenta al sistema.

Nótese que cuando usa el comodín "*", no se muestra ninguno de los nombres de fichero que comienzan por ".".

```
/home/admin# ls *
frog    cnt    stuff
/home/admin#
```

Esto es una característica de seguridad: si "*" coincidiera con ficheros que comienzan por ".", actuaría sobre "." y "..". Esto puede ser peligroso con ciertas órdenes.

Otro carácter comodín es "?". Este carácter comodín solo expande un único carácter. Luego "ls?" mostrará todos los nombres de ficheros con un carácter de longitud, y "ls termca?" mostrara "termcap" pero no "termcap.backup". A continuación otro ejemplo:

```
/home/admin# ls c?t
cnt
/home/admin# ls f??g
frog
/home/admin# ls ????f
stuff
/home/admin#
```

Como puede ver, los caracteres comodines permiten referirse a más de un fichero a la vez. En el resumen de ordenes en la Sección 11 se dijo que cp y mv pueden copiar o mover múltiples ficheros de una vez. Por ejemplo,

```
/home/admin# cp /etc/s* /home/admin
```

Este comando copiará todos los ficheros de /etc que comiencen por "s" al directorio /home/admin. Por lo tanto, el formato de la orden cp es realmente

```
cp <fichero1> <fichero2> <fichero3> ...<ficheroN> <destino>
```

donde <fichero1> a <ficheroN> es la lista de los ficheros a copiar, y <destino> es el fichero o directorio destino donde copiarlos. mv tiene idéntica sintaxis.

Nótese que si está copiando o moviendo más de un fichero, <destino> debe ser un directorio. Solo puede copiar o mover un único fichero a otro fichero.

14. Fontanería Linux

14.1 Entrada y salida estándar

Muchos comandos UNIX toman su entrada de algo conocido como “entrada estándar” y envían su salida a la “salida estándar” (a menudo abreviado como "stdin" y "stdout"). El

intérprete de comandos configura el sistema de forma que la entrada estándar es el teclado y la salida la pantalla.

Veamos un ejemplo con el comando cat. Normalmente cat lee datos de los ficheros cuyos nombres se pasan como argumentos en la línea de comandos y envía estos datos directamente a la salida estándar. Luego, se usa el comando

/home/admin/papers# cat history-final masters-thesis

mostrará en pantalla el contenido del fichero history-final seguido por masters-thesis.

Si no se le entregan nombres de ficheros a cat como parámetros, leerá datos de stdin y los enviará a stdout. Vea el siguiente ejemplo.

/home/admin/papers# cat
Hello there.
Hello there.
Bye.
Bye.

“ctrl-D”
/home/admin/papers#

Como se puede ver, cada línea que el usuario teclea (impresa en itálica) es inmediatamente reenviada al monitor por cat. Cuando se está leyendo de la entrada estándar, los comandos reconocen el fin de la entrada de datos cuando reciben el carácter EOT (end-of-text, fin de texto). Normalmente es generado con la combinación “ctrl-D”.

Vea otro ejemplo. El comando sort toma como entrada líneas de texto (de nuevo leerá desde stdin si no se le proporcionan nombres de ficheros en la línea de comandos), y devuelve la salida ordenada a stdout.

/home/admin/papers# sort
platanos
zanahorias

manzanas
“ctrl-D”
manzanas
platanos
zanahorias
/home/admin/papers#

Puede ordenar alfabéticamente la lista de la compra... ¿no es útil Linux?

14.2 Redireccionando la entrada y salida

Ahora, suponga que quiere que la salida de sort vaya a un fichero para poder salvar la lista ordenada de salida. El interprete de comandos le permitirá redireccionar la salida estándar a un fichero usando el símbolo ">". Vea como funciona.

/home/admin/papers# sort > lista-compras
platanos
zanahorias
manzanas

“Ctrl-D”
/home/admin/papers#

Como puede ver, el resultado de sort no se muestra por pantalla, en su lugar es salvado en el fichero lista-compras. Ahora mire el fichero.

/home/admin/papers# cat lista-compras
manzanas
platanos
zanahorias
/home/admin/papers#

Ya puede ordenar la lista de la compra y además guardarla.

Supongamos ahora que tiene guardada su lista de compra desordenada en el fichero ítems. Una forma de ordenar la información y salvarla en un fichero podría ser darle a sort el nombre del fichero a leer en lugar de la entrada estándar y redireccionar la salida estándar como se hizo anteriormente.

```
/home/admin/papers# sort ítems > lista-compras
/home/admin/papers# cat lista-compras
manzanas
plátanos
zanahorias
/home/admin/papers#
```

Hay otra forma de hacer esto. No solo puede ser re-direccionada la salida estándar, también puede ser re-direccionada la entrada estándar usando el símbolo "<".

```
/home/admin/papers# sort < ítems
manzanas
platanos
zanahorias
/home/admin/papers#
```

Técnicamente, sort < ítems es equivalente a sort ítems, pero permite demostrar que sort < ítems se comporta como si los datos del fichero fueran tecleados por la entrada estándar. El intérprete de comandos es quien maneja las re-direcciones. sort no recibe el nombre del fichero (ítems) a leer, desde el punto de vista de sort, esta leyendo datos de la entrada estándar como si fueran tecleados.

Esto introduce el concepto de filtro. Un filtro es un programa que lee datos de la entrada estándar, los procesa de alguna forma, y devuelve los datos procesados por la salida estándar. Usando la redirección la entrada estándar y/o salida estándar pueden ser referenciadas desde ficheros. Sort es un filtro simple: ordena los datos de entrada y envía el resultado a la salida estándar, cat es incluso más simple, no manipula los datos de entrada, simplemente envía a la salida cualquier cosa que le llega.

14.3 Uso de tuberías (pipes)

Ya se ha visto como usar sort como un filtro. Pero estos ejemplos suponen que se tienen los datos en un fichero en alguna parte o va a introducir los datos manualmente por la entrada estándar.

¿Que pasa si los datos que quiere ordenar provienen de la salida de otro comando, como ls?. Por ejemplo, usando la opción -r con sort se ordenan los datos en forma inversa. Si quiere listar los ficheros en el directorio actual en orden inverso, una forma podría ser.

```
/home/admin/papers# ls
english-list
history-final
masters-thesis
notes
/home/admin/papers# ls > file-list
/home/admin/papers# sort -r file-list
notes
masters-thesis
history-final
english-list
/home/admin/papers#
```

Aquí, se guarda la salida de ls en un fichero, y entonces se ejecuta sort -r sobre ese fichero. Pero esta forma necesita crear un fichero temporal en el que se guardarán los datos generados por ls.

La solución es usar los pipes[9]. El uso de pipes es otra característica del intérprete de comandos, que permite conectar una cadena de comandos en un "pipe", donde la stdout del primero es enviada directamente a la stdin del segundo y así sucesivamente. Se quiere conectar la salida de ls con la entrada de sort. Para crear un pipe se usa el símbolo "|":

[9] Tuberías

```
/home/admin/papers# ls | sort -r
notes
masters-thesis
history-final
english-list
/home/admin/papers#
```

Esta forma es más corta y obviamente más fácil de escribir. Otro ejemplo útil usando el comando:

```
/home/admin/papers# ls /usr/bin
```

Mostrará una larga lista de los ficheros, la mayoría de los cuales pasará rápidamente ante sus ojos sin que pueda leerla. En lugar de esto, use *more* para mostrar la lista de ficheros en /usr/bin.

```
/home/admin/papers# ls /usr/bin | more
```

Ahora puede ir avanzando página a página cómodamente, puede además "entubar" más de dos comandos a la vez. El comando head es un filtro que muestra las primeras líneas del canal de entrada (aquí la entrada desde una pipe). Si quiere ver el último fichero del directorio actual en orden alfabético, se usa:

```
/home/admin/papers# ls | sort -r | head -1
notes
/home/admin/papers#
```

Donde head -1 simplemente muestra la primera línea de la entrada que recibe en este caso, el flujo de datos ordenados inversamente provenientes de ls.

14.4 Redirección no destructiva

El uso de ">" para redireccionar la salida a un fichero es destructivo: en otras palabras, el comando

/home/admin/papers# ls > file-list

Sobreescribe el contenido del fichero file-list. Si en su lugar, usa el símbolo ">>", la salida será añadida al final del fichero nombrado, en lugar de ser sobrescrito.

/home/admin/papers# ls >> file-list

Añadirá la salida de ls al final de file-list.

Es conveniente tener en cuenta que la redirección y el uso de pipes son características proporcionadas por el intérprete de comandos, este, proporciona estos servicios mediante el uso de la sintaxis ">", ">>" y "|".

15. Permisos de Ficheros

15.1 Conceptos de permisos de ficheros

Al ser Linux un sistema multiusuario, para proteger ficheros de usuarios particulares de la manipulación por parte de otros, Linux proporciona un mecanismo conocido como permisos de ficheros. Este mecanismo permite que ficheros y directorios "pertenezcan" a un usuario en particular. Por ejemplo, como Admin creó ficheros en su directorio "home", Admin es el propietario de esos ficheros y tiene acceso a ellos.

Linux también permite que los ficheros sean compartidos entre usuarios y grupos de usuarios. Si Admin lo desea, podría restringir el acceso a sus ficheros de forma que ningún otro usuario tenga acceso. De cualquier modo, en la mayoría de los sistemas por defecto se permite que otros usuarios puedan leer sus ficheros pero no modificarlos o borrarlos.

Como se ha explicado, cada fichero pertenece a un usuario en particular. Por otra parte, los ficheros también pertenecen a un grupo en particular, que es un conjunto de usuarios definido por el sistema. Cada usuario pertenece al menos a un grupo cuando es creado. El administrador del sistema puede hacer que un usuario tenga acceso a más de un grupo. Los grupos usualmente son definidos por el tipo de usuarios que acceden a la máquina. Por ejemplo, en un sistema Linux de una Universidad, los usuarios pueden ser divididos en los grupos estudiantes, dirección, profesores e invitados. Hay también unos pocos grupos definidos por el sistema (como bin y admin) los cuales son usados por el propio sistema para controlar el acceso a los recursos, muy raramente los usuarios normales pertenecen a estos grupos.

Los permisos están divididos en tres tipos: lectura, escritura y ejecución. Estos permisos pueden ser fijados para tres clases de usuarios: el propietario del fichero, el grupo al que pertenece el fichero y para todos los usuarios independientemente del grupo.

El permiso de lectura permite a un usuario leer el contenido del fichero o en el caso de un directorio, listar el contenido del mismo (usando ls). El permiso de escritura permite a un usuario escribir y modificar el fichero. Para directorios, el permiso de escritura permite crear nuevos ficheros o borrar ficheros ya existentes en dicho directorio. Por último, el permiso de ejecución permite a un usuario ejecutar el fichero si es un programa o guión del intérprete de comandos. Para directorios, el permiso de ejecución permite al usuario cambiar al directorio en cuestión con cd.

15.2 Interpretando los permisos de ficheros

Analice un ejemplo del uso de permisos de ficheros. Usando el comando ls con la opción -l se mostrará un listado "largo" de los ficheros, el cual incluye los permisos de ficheros.

```
/home/admin/foo# ls -l stuff

-rw-r--r--    1 admin     users         505 Mar 13 19:05 stuff

/home/admin/foo#
```

El primer campo impreso en el listado representa los permisos de ficheros. El tercer campo es el propietario del fichero (admin), y el cuarto es el grupo al cual pertenece el fichero (users). Obviamente, el último campo es el nombre del fichero (stuff), y los demás campos se tratarán mas adelante.

Este fichero pertenece a admin y al grupo users. Vea los permisos dados. La cadena -rw-r--r-- nos informa, por orden, de los permisos para el propietario, el grupo del fichero y cualquier otro usuario.

El primer carácter de la cadena de permisos ("-") representa el tipo de fichero. El "-" significa que es un fichero regular. Las siguientes tres letras ("rw-") representan los permisos para el propietario del fichero, admin. El "r" para "lectura" y "w" para escritura. Luego Admin tiene permisos de lectura y escritura para el fichero stuff.

Como ya se mencionó, aparte de los permisos de lectura y escritura esta el permiso de "ejecución", representado por una "x". Como hay un "-" en lugar del "x", significa que Admin no tiene permiso para ejecutar ese fichero. Esto es correcto, puesto que stuff no es un programa de ningún tipo. Por supuesto, como el fichero es de Admin, puede darse a si mismo permiso de ejecución si lo desea. Esto será tratado en breve. Los siguientes tres caracteres, r-- representan los permisos para los miembros del grupo. El grupo al que pertenece el fichero es users. Como solo aparece un "r" cualquier usuario que pertenezca al grupo users puede leer este fichero. Los últimos tres caracteres, también r--, representan los permisos para cualquier otro usuario del sistema (diferentes del propietario o de los pertenecientes al grupo users). De nuevo, como solo esta presente el "r", los demás usuarios pueden leer el fichero, pero no escribir en el o ejecutarlo.

Otros ejemplos de permisos de grupo.

-rwxr-xr-x	El propietario del fichero puede leer, escribir y ejecutar el fichero. Los usuarios pertenecientes al grupo del fichero, y todos los demás usuarios pueden leer y ejecutar el fichero.
-rw-------	El propietario del fichero puede leer y escribir. Nadie más puede acceder al fichero.
-rwxrwxrwx	Todos los usuarios pueden leer, escribir y ejecutar el fichero.

15.3 Dependencias

Es importante darse cuenta de que los permisos de un fichero también dependen de los permisos del directorio en el que residen. Por ejemplo, aunque un fichero tenga los permisos -rwxrwxrwx, otros usuarios no podrán acceder a el a menos que también tengan permiso de lectura y ejecución para el directorio en el cual se encuentra el fichero. Si Admin quiere restringir el acceso a todos sus ficheros, podría simplemente poner los permisos de su directorio "home" /home/admin a -rwx------. De esta forma ningún usuario podrá acceder a su directorio ni a ninguno de sus ficheros o subdirectorios. Admin no necesita preocuparse de los permisos individuales de cada uno de sus ficheros.

En otras palabras, para acceder a un fichero, se debe tener permiso de ejecución de todos los directorios a lo largo del camino de acceso al fichero, además de permiso de lectura (o ejecución) del fichero en particular.

Habitualmente, los usuarios de un sistema Linux son muy abiertos con sus ficheros. Los permisos que se dan a los ficheros usualmente son -rw-r--r--, lo que permite a todos los demás usuarios leer los ficheros, pero no modificarlos de ninguna forma. Los directorios, usualmente tienen los permisos -rwxr-xr-x, lo que permite que los demás usuarios puedan moverse y ver los directorios, pero sin poder crear o borrar nuevos ficheros en ellos.

Muchos usuarios pueden querer limitar el acceso de otros usuarios a sus ficheros, colocando los permisos de un fichero a -rw------- no se permitirá a ningún otro usuario acceder al fichero.

Igualmente, poniendo los permisos del directorio a -rwx------ no se permitirá a los demás usuarios acceder al directorio en cuestión.

15.4 Cambiando permisos

El comando chmod se usa para establecer los permisos de un fichero. Solo el propietario puede cambiar los permisos del fichero. La sintaxis de chmod es:

```
chmod {a,u,g,o}{+,-}{r,w,x} <filenames>
```

El comando indica a que usuarios afecta: (a) all, (u) user, (g) group o (o) other. Entonces se especifica si se están añadiendo permisos (+) o quitándolos (-). Finalmente se especifica que tipo de permiso (r) Read, (w) write o (x) execute. Algunos ejemplos:

chmod a+r stuff: Da a todos los usuarios acceso al fichero.

chmod +r stuff: Como arriba, si no se indica a, u, g, o por defecto se toma a.

chmod og-x stuff: Quita permisos de ejecución a todos los usuarios excepto al propietario.

chmod u+rwx stuff: Permite al propietario leer, escribir y ejecutar el fichero.

chmod o-rwx stuff: Quita permisos de lectura, escritura y ejecución a todos los usuarios menos al propietario y a los usuarios del grupo del fichero.

16. Manejando enlaces de ficheros

Los enlaces permiten dar a un único fichero múltiples nombres. Los ficheros son identificados por el sistema por su número de inodo, el cual es el único identificador del fichero para el sistema de ficheros[10]. Un directorio es una lista de números de inodo con sus correspondientes nombres de fichero. Cada nombre de fichero en un directorio es un enlace a un inodo particular.

16.1 Enlaces duros (Hard links)

La orden ln es usada para crear múltiples enlaces para un fichero. Por ejemplo, suponga que se tiene un fichero foo en un directorio. Usando ls -i, verá el número de inodo para el fichero.

```
# ls -i foo
22192 foo
```

[10] La orden ls -i mostrara los números de inodo.

Aquí, el fichero foo tiene el número de inodo 22192 en el sistema de ficheros. Puede crear otro enlace a foo, llamado bar:

```
# ln foo bar
```

Con ls -i verá que los dos ficheros tienen el mismo inodo.

```
# ls -i foo bar
22192 bar    22192 foo
#
```

Ahora, accediendo a foo o a bar accederá al mismo fichero. Si hace cambios en foo, estos cambios también serán efectuados en bar. Para todos los efectos, foo y bar son el mismo fichero.

Estos enlaces son conocidos como enlaces duros (Hard links) porque directamente crean el enlace al inodo. Nótese que sólo puede crear enlaces duros entre ficheros del mismo sistema de ficheros; enlaces simbólicos (ver mas adelante) no tienen esta restricción.

Cuando borra un fichero con rm, está solamente borrando un enlace a un fichero. Si usa el comando

```
# rm foo
```

Sólo el enlace de nombre foo es borrado; bar todavía existirá. Un fichero es definitivamente borrado del sistema cuando no quedan enlaces a él. Usualmente, los ficheros tienen un único enlace, por lo que el uso de rm los borra. Pero si el fichero tiene múltiples enlaces, el uso de rm solo borrará un único enlace; para borrar el fichero, debemos borrar todos los enlaces del fichero.

La orden ls -l muestra el número de enlaces a un fichero (entre otra información).

```
# ls -l foo bar
-rw-r--r--   2 root     root          12 Aug   5 16:51 bar
-rw-r--r--   2 root     root          12 Aug   5 16:50 foo
#
```

La segunda columna en el listado, "2", especifica el numero de enlaces al fichero.

Así resulta que un directorio no es más que un fichero que contiene información sobre la translación enlace a inodo. También, cada directorio tiene al menos dos enlaces duros en él: "." (un enlace apuntando a si mismo) y ".." (un enlace apuntando al directorio padre). En el directorio raíz (/), el enlace ".." simplemente apunta a /.

16.2 Enlaces simbólicos

Los enlaces simbólicos son otro tipo de enlace, que es diferente al enlace duro. Un enlace simbólico permite dar a un fichero el nombre de otro, pero no enlaza el fichero con un inodo. La orden ln -s crea un enlace simbólico a un fichero. Por ejemplo, si usa la orden

```
# ln -s foo bar
```

Creará un enlace simbólico bar apuntando al fichero foo. Si usa ls -i, verá que los dos ficheros tienen inodos diferentes, en efecto.

```
# ls -i foo bar
22195 bar    22192 foo
#
```

De cualquier modo, usando ls -l ve que el fichero bar es un enlace simbólico apuntando a foo.

```
# ls -l foo bar
lrwxrwxrwx        1 root     root           3 Aug   5 16:51 bar -> foo
-rw-r--r--        1 root     root          12 Aug   5 16:50 foo
#
```

Los bits de permisos en un enlace simbólico no se usan (siempre aparecen como rwxrwxrwx). En su lugar, los permisos del enlace simbólico son determinados por los permisos del fichero "apuntado" por el enlace (para éste ejemplo, el fichero foo).

Funcionalmente, los enlaces duros y simbólicos son similares, pero hay algunas diferencias. Por una parte, puede crear un enlace simbólico a un fichero que no existe; lo mismo no es aplicable para enlaces duros. Los enlaces simbólicos son procesados por el núcleo de forma diferente a los duros, lo cual es sólo una diferencia técnica, pero a veces importante. Los enlaces simbólicos son de ayuda puesto que identifican al fichero al que apuntan; con enlaces duros no hay forma fácil de saber que fichero esta enlazado al mismo inodo. Los enlaces se usan en muchas partes del sistema Linux. Los enlaces simbólicos son especialmente importantes para las imágenes de las librerías compartidas en /lib.

17. Tareas y procesos

Control de Tareas es una utilidad incluida en muchos shells (incluidas Bash y Tcsh), que permite el control de multitud de comandos o tareas al momento. Antes de seguir, se hablará sobre los procesos.

Cada vez que se ejecuta un programa, se lanza lo que se conoce como proceso, que es simplemente el nombre que se le da a un programa cuando se esta ejecutando. El comando ps visualiza la lista de procesos que se están ejecutando actualmente, por ejemplo:

```
/home/admin# ps
  PID TT STAT   TIME COMMAND
   24   3 S      0:03 (bash)
  161   3 R      0:00 ps
/home/admin#
```

La columna PID representa el identificador de proceso. La última columna COMMAND, es el nombre del proceso que se está ejecutando. Ahora está viendo los procesos que está ejecutando Admin[11]. Vea que hay dos procesos, bash (Que es el shell o intérprete de comandos que usa Admin), y el propio comando ps. Como se puede observar, bash se ejecuta concurrentemente con el comando ps. Bash ejecutó ps cuando Admin tecleó el comando. Cuando ps termina de ejecutarse (después de mostrar la tabla de procesos), el control retorna al proceso bash, que muestra el prompt, indicando que está listo para recibir otro comando.

Un proceso que está corriendo se denomina tarea para el shell. Los términos proceso y tarea, son intercambiables. Sin embargo, se suele denominar "tarea" a un proceso, cuando es usado en conjunción con control de tareas, que es un rasgo del shell que permite cambiar entre distintas tareas.

En muchos casos, los usuarios sólo ejecutan un trabajo cada vez, que es el último comando que ellos teclearon desde el shell. Sin embargo, usando el control de tareas, se puede ejecutar diferentes tareas al mismo tiempo, cambiando entre cada uno de ellos conforme lo vaya necesitando. ¿Cuan beneficioso puede llegar a ser esto?. Suponga que esta trabajando con un procesador de textos, y de repente necesita parar y realizar otra tarea, con el control de tareas, podría suspender temporalmente el editor, y volver al shell para realizar cualquier otra tarea, y luego regresar al editor como si no lo hubiese dejado nunca. Lo siguiente solo es un ejemplo, hay montones de usos prácticos del control de tareas.

17.1 Primer plano y Segundo plano

Un proceso puede estar en Primer plano o en Segundo plano. Solo puede haber un proceso en primer plano al mismo tiempo, el proceso que está en primer plano, es el que interactúa con usted recibiendo entradas de teclado, y envía las salidas al monitor. El proceso en segundo plano, no recibe ninguna señal desde el teclado, por lo general, se ejecutan en silencio sin necesidad de interacción.

Algunos programas necesitan mucho tiempo para terminar, y no hacen nada interesante mientras tanto. Compilar programas es una de estas tareas, así como comprimir un fichero grande. No tiene sentido perder tiempo mientras estos procesos terminan. En estos casos es

[11] Hay muchos más procesos aparte de estos corriendo en el sistema, para verlos todos

mejor lanzarlos en segundo plano, para dejar al computador en condiciones de ejecutar otro programa.

Los procesos pueden ser suspendidos. Un proceso suspendido es aquel que no se está ejecutando actualmente, sino que está temporalmente detenido. Después de suspender una tarea, puede indicar a la misma que continúe, en primer o en segundo plano, según necesitemos retomar una tarea suspendida no cambia en nada el estado de la misma, continuará ejecutándose justo donde se dejó.

Es necesario tener en cuenta que suspender un trabajo no es lo mismo que interrumpirlo. Cuando se interrumpe un proceso (generalmente con la pulsación de “ctrl-C”), el proceso muere, y deja de estar en memoria sin utilizar recursos del computador. Una vez eliminado, el proceso no puede continuar ejecutándose, y deberá ser lanzado otra vez para volver a realizar sus tareas. También se puede dar el caso de que algunos programas capturan la interrupción, de modo que pulsando “ctrl-C” |no se detengan inmediatamente. Esto se hace para permitir al programa realizar operaciones necesarias de limpieza antes de terminar. De hecho, algunos programas simplemente no se dejan matar por ninguna interrupción.

17.2 Envío a segundo plano y eliminación de procesos

Se comenzará con un ejemplo sencillo. El comando “yes” es un comando aparentemente inútil que envía una serie interminable de íes (y) a la salida estándar. (Realmente es muy útil, si se utiliza una tubería o "pipe" para unir la salida de yes con otro comando que haga preguntas del tipo si/no, la serie de íes confirmará todas las preguntas.)

Ejemplo.

```
/home/admin# yes
y
y
y
y
```

La serie de íes continuará hasta el infinito, a no ser que se elimine, pulsando la tecla de interrupción, generalmente “ctrl-C”. También puede deshacerse de esta serie de íes redigiriendo

la salida estándar de “yes” hacia /dev/null, que como se vio anteriormente es una especie de "agujero negro" o papelera para los datos. Todo lo que se envíe allí, desaparecerá.

```
/home/admin# yes > /dev/null
```

Ahora va mucho mejor, el terminal no se ensucia, pero el prompt de la shell no retorna. Esto es porque yes sigue ejecutándose y enviando esos inútiles íes a /dev/null. Para recuperarlo, se pulsa la tecla de interrupción.

Suponga ahora que quiere dejar que el comando yes siga ejecutándose, y volver al mismo tiempo a la shell para trabajar en otras cosas. Para ello debe enviar a yes a segundo plano, lo que le permitirá ejecutarlo, pero sin necesidad de interacción.

Una forma de mandar procesos a segundo plano es añadiendo un carácter "&" al final de cada comando.

```
/home/admin# yes > /dev/null &
[1] 164
/home/admin#
```

Como se puede ver, ha regresado al shell. Pero, ¿qué es eso de "[1] 164"?, ¿se está ejecutando realmente el comando yes?

"[1]" representa el número de tarea del proceso yes. Shell asigna un número a cada tarea que se esté ejecutando. Como yes es el único comando que se está ejecutando, se le asigna el número de tarea 1. El número "164" es el número de identificación del proceso, o PID, que es el número que el sistema le asigna al proceso. Ambos números pueden usarse para referirse a la tarea como se verá después.

Ahora tiene el proceso yes corriendo en segundo plano, y enviando constantemente la señal y hacia el dispositivo /dev/null. Para chequear el estado del proceso, use el comando interno de la shell jobs:

```
/home/admin# jobs
[1]+  Running                 yes >/dev/null   &
/home/admin#
```

También puede usar el comando ps, como se mostró anteriormente, para comprobar el estado de la tarea. Para eliminar una tarea, use el comando kill. Este comando toma como argumento un número de tarea o un número de ID de un proceso. Esta era la tarea N° 1, así que usando el comando

```
/home/admin# kill %1
```

matará la tarea. Cuando se identifica la tarea con el número, se debe preceder el número con el carácter de porcentaje (`%").
Ahora que ya ha matado la tarea, puede usar el comando jobs de nuevo para comprobarlo:

```
/home/admin# jobs

[1]+  Terminated              yes >/dev/null

/home/admin#
```

La tarea está, en efecto, muerta, y si usa el comando jobs de nuevo, no mostrará nada. También puede matar la tarea usando el numero de ID de proceso (PID), el cual se muestra conjuntamente con el ID de tarea cuando arranca la misma. En el ejemplo, el ID de proceso es 164, así que el comando será

```
/home/admin# kill 164
```

es equivalente a

```
/home/admin# kill %1
```

No es necesario usar el "%" cuando se refiere a una tarea a través de su ID de proceso.

17.3 Parada y relanzamiento de tareas

Hay otra manera de poner una tarea en segundo plano. Se puede lanzar como un proceso normal (en primer plano), pararlo, y después relanzarlo en segundo plano.

Primero, lance el proceso yes en primer plano como se haría normalmente:

```
/home/admin# yes > /dev/null
```

De nuevo, dado que yes corre en primer plano, no se debe retornar el prompt de la shell. Ahora, en vez de interrumpir la tarea con “ctrl-C”, suspenda la tarea. El suspender una tarea no la mata; solamente la detiene temporalmente hasta que quiera retomarla. Para hacer esto se debe pulsar la tecla de suspender, que suele ser “ctrl-Z”.

```
/home/admin#_yes > /dev/null

“ctrl-Z”
[1]+  Stopped                 yes >/dev/null
/home/admin#
```

Mientras el proceso está suspendido, simplemente no se está ejecutando. No gasta tiempo de CPU en la tarea. Sin embargo, puede retomar el proceso de nuevo como si nada hubiese pasado, continuará ejecutándose donde se dejó.

Para relanzar la tarea en primer plano, use el comando fg (del inglés "foreground").

```
/home/admin# fg
yes >/dev/null
```

El Shell muestra el nombre del comando de nuevo, de forma que tenga conocimiento de que tarea es la que ha puesto en primer plano. Pare la tarea de nuevo, con “ctrl-Z”, Esta vez utilice el comando bg para poner la tarea en segundo plano. Esto hará que el comando siga ejecutándose igual que si lo hubiese hecho desde el principio con "&" como en la sección anterior.

```
/home/admin# bg
[1]+ yes >/dev/null &
/home/admin#
```

Y tendrá de nuevo el prompt. El comando jobs debería decirle que yes se está ejecutando, y puede matar la tarea con kill tal y como se hizo anteriormente.

¿Como puede parar la tarea de nuevo? Si pulsa “ctrl-Z”, no funcionará, ya que el proceso está en segundo plano. La respuesta es poner el proceso en primer plano de nuevo, con el comando fg y entonces pararlo. Como se puede observar podrá usar fg tanto con tareas detenidas, como con las que estén en segundo plano.

Hay una gran diferencia entre una tarea que se encuentra en segundo plano, y una que se encuentra detenida. Una tarea detenida es una tarea que no se está ejecutando, es decir, que no usa tiempo de CPU, y que no está haciendo ningún trabajo (la tarea aún ocupa un lugar en memoria, aunque puede ser volcada a disco). Una tarea en segundo plano, se está ejecutando, y usando memoria, a la vez que está completando alguna acción mientras hace otro trabajo. Sin embargo, una tarea en segundo plano puede intentar mostrar texto en el terminal, lo que puede resultar molesto si está intentando hacer otra cosa. Por ejemplo, si usa el comando

```
/home/admin# yes &
```

sin redirigir stdout a /dev/null, una cadena de íes se mostrarán en el monitor, sin modo alguno de interrumpirlo (no puede hacer uso de “ctrl-C” para interrumpir tareas en segundo plano). Para poder parar esas interminables íes, tendrá que usar el comando fg para pasar la tarea a primer plano, y entonces usar “ctrl-C” para matarla.

Otra observación, normalmente, los comandos "fg" y "bg" actúan sobre el ultimo proceso parado (indicado por un "+" junto al numero de tarea cuando usa el comando jobs). Si se tienen varios procesos corriendo a la vez, se puede mandar a primer o segundo plano una tarea específica indicando el ID de tarea como argumento de fg o bg, como en

```
/home/admin# fg %2
```

(para la tarea de primer plano número 2), o

```
/home/admin# bg %3
```

(para la tarea de segundo plano número 3). No se pueden usar los ID de proceso con fg o bg. Además de esto, si usa el número de tarea por si solo, como:

```
/home/admin# %2
```

es equivalente a:

```
/home/admin# fg %2
```

Es necesario tener en cuenta que el uso del control de tareas es una utilidad del shell. Los comandos fg, bg y jobs son internos del shell. Si por algún motivo se utiliza una shell que no soporta control de tareas, no dispondremos de estos comandos. Y además, hay algunos aspectos del control de tareas que difieren entre Bash y Tcsh. De hecho, algunos shells no proporcionan ningún control de tareas sin embargo, la mayoría de ellos, disponibles para Linux soportan control de tareas.

CAPITULO V
SERVICIOS

18. Apache

Apache es el servidor web de libre distribución más usado. Se encuentra incluido en casi la totalidad de versiones de UNIX e incluso existen versiones para Windows. Su distribución en base a código fuente hace posible su compilación bajo casi cualquier sistema operativo o arquitectura.

La configuración del servidor Apache se define en tres ficheros diferentes:

a) httpd.conf: Que contiene información sobre el modo de ejecución del servidor, usuario, permisos, y directorios de log, documentos, etc.
b) srm.conf: Asociado a los mecanismos de presentación de documentos, la ejecución de CGIs y otras configuraciones similares.
c) access.conf: Detalla los permisos y las directivas de acceso a diferentes ramas del árbol de documentos.

Estos ficheros se encuentran habitualmente en el directorio /etc/httpd/ en la mayoría de las distribuciones de Linux, el cual instala por omisión este servidor web.

18.1 El fichero httpd.conf

Los parámetros más relevantes de este fichero de configuración son:

User *nombre-de-usuario*
User #*uid-de-usuario*

Usuario que arranca el servicio (habitualmente nobody). Este usuario es el propietario del proceso httpd en ejecución.

Group *nombre-de-grupo*
Group #*gid-de-grupo*
Idem para el grupo.

ServerName *nombre-del-servidor*
Para computadoras con varias entradas de nombre en el servidor DNS, esta directiva determina cual es el nombre canónico usado para las peticiones HTTP.

ServerAdmin *dirección-de-correo*
Dirección del correo del administrador del servicio web.

ServerRoot *ruta-de-directorio*
Especifica cuál es la ruta del directorio raíz a partir del cual se definen el resto de directorios de configuración y log.

PidFile *ruta-de-fichero*
Ruta relativa al ServerRoot donde se encuentra el fichero que contiene el PID del proceso httpd.

ErrorLog *ruta-de-fichero*
Ruta relativa al ServerRoot donde se encuentra el fichero que contiene los errores producidos durante la ejecución del servidor.

TransferLog *ruta-de-fichero*
Ruta relativa al ServerRoot donde se encuentra el fichero que registra todas las transferencias de documentos solicitadas al servidor.

HostnameLookups [on | off]
Indica si en las entradas del TransferLog se indican las direcciones IP de los clientes o si se realiza una búsqueda de sus nombres simbólicos asociados.

CacheNegotiateDocs
Especifica si los documentos almacenados en el servidor pueden ser almacenados, a su vez, por los *proxies* que lo consultan.

KeepAlive *número-de-peticiones*
Indica el número de peticiones para las cuales se mantiene una misma conexión abierta. Esta opción optimiza el número de conexiones usadas para cargar una página HTML.

KeepAliveTimeout *segundos*
Define el tiempo que ha de pasar para cerrar una conexión abierta para la transmisión de varias peticiones en el caso de que no se hagan todas ellas.

ServerType [inetd | standalone]
Indica si el servidor web es arrancado como un servicio por el demonio inetd o si por el contrario es un servidor que esta en ejecución por sí mismo. En el segundo caso, es necesario determinar ciertos parámetros indicados por las directivas siguientes.

Port *número-de-puerto*
Especifica el puerto de servicio, usualmente se trata del puerto 80.

StartServers *número-de-servidores*
Indica cuantas copias del servidor web son arrancadas al iniciarse el servicio.

MinSpareServers *número-de-servidores*
MaxSpareServers *número-de-servidores*
Estas dos directivas determinan cuántos servidores web se encuentran arrancados y sin atender peticiones. Estos dos parámetros permiten que el servidor web gestione dinámicamente la carga.

MaxClients *número-de-clientes*
Especifica el número máximo de clientes concurrentes que el servicio puede atender.

MaxRequestPerChild *número-de-peticiones*
Indica el número máximo que peticiones que una misma copia del servidor puede realizar.

TimeOut *segundos*
Indica el tiempo máximo que una copia del servidor puede permanecer inactiva sin servir peticiones.

18.2 El fichero srm.conf

DocumentRoot *ruta-de-directorio*
Indica el directorio a partir del cual se encuentran las páginas HTML del servidor. Todas las peticiones web al servidor se resuelven usando como referencia a este directorio del servidor.

UserDir *ruta-relativa-de-directorio*
Determina la ruta relativa a cada directorio de usuario usada para resolver las peticiones del tipo: http://servidor.dominio/~usuario/

DirectoryIndex *nombres-de-ficheros*
Indica los nombres de ficheros usados como posibles páginas de referencia cuando la petición es realizada sobre un directorio.

FancyIndexing [on | off]
Habilita la presentación alternativa de los contenidos de un directorio si no existe la página de índice.

ReadmeName *nombre-de-fichero*
HeaderName *nombre-de-fichero*
Estas dos directivas determinan el texto que se incluye entes (HeaderName) y después (ReadmeName) del contenido del directorio.

IndexIgnore *nombres-de-ficheros*

Indica los ficheros que no son presentados a la hora de mostrar el contenido de un directorio.

AddIcon *fichero-de-icono extensión*
AddIconByType *fichero-de-icono tipo-mime*
AddIconByEncoding *fichero-de-icono tipo-mime-encoding*
AddDescription *texto extensión*

Estas directivas se utilizan para definir la presentación del contenido de un directorio en base a los tipos y extensiones de ficheros que ahí se encuentran.

Alias *ruta-de-directorio-web ruta-de-directorio*
Esta opción permite asociar a un directorio del servicio web un directorio real determinado. Esta opción puede referenciar directorios fuera del árbol definido por DocumentRoot.

ScriptAlias *ruta-de-directorio-web ruta-de-directorio*
Esta opción es idéntica a la anterior, pero el directorio web definido es usado únicamente para ejecutables CGI. De este directorio no es visible su contenido.

18.3 El fichero access.conf

Este fichero define las políticas de acceso y seguridad de los documentos del servidor. Estas restricciones pueden definirse con ámbito general en este fichero o localmente en cada directorio por medio de un fichero de configuración específico del directorio.

AccessFileName *nombre-de-fichero*
Indica el nombre del fichero de seguridad local de cada directorio.

<Limit *métodos*>
directivas
</Limit>

Este bloque se usa para definir las premisas de acceso y consulta a un directorio determinado. Los métodos posibles son: GET, HEAD, POST, PUT y DELETE. Las directivas que pueden aparecer dentro de este bloque son, por ejemplo:

```
order deny, allow
deny from all
allow from elei.uach.cl
```

Estas directivas restringen el acceso a todas las maquinas fuera del dominio elei.uach.cl. Junto con las directivas de acceso basadas en direcciones de cliente, es posible también definir restricciones de acceso basadas en usuarios y palabras clave. Un ejemplo de estas restricciones es:

```
AuthUserFile /etc/httpd/http_users
AuthName     “Servidor Departamental”
AuthType     Basic
```

Este ejemplo restringe el acceso por medio de un usuario y clave definidos en el fichero /etc/httpd/http_users. La ventana de petición mostrara el mensaje “*Servidor Departamental*” y la verificación de las palabras clave se realiza por el algoritmo habitual de UNIX (La opción Digest habilita la verificación vía MD5).

18.4 Gestión del Servidor

El arranque y la parada del servidor se realizan, dependiendo de la instalación, por medio del *script* de administración:

```
/etc/rc.d/init.d/httpd     o
/etc/init.d/httpd
```

Cualquier cambio en la configuración del servidor requiere rearrancar el servicio, esto se puede realizar por medio de una serie de argumentos proporcionados al *script*. Estos argumentos son:

start: Arranca el servicio, si este se encontrase detenido.
stop: Detiene el servicio si estaba en ejecución.
restart: Detiene y arranca de nuevo el servicio.
status: Muestra información relativa al estado del servidor (en ejecución o detenido).

19. DNS

DNS (Domain Name Service) organiza los nombres de máquina (hostname) en una jerarquía de dominios. Un *dominio* es una colección de nodos relacionados de alguna forma—porque estén en la misma red, tal como los nodos de una universidad—. Por ejemplo, el dominio que agrupa a la Universidad Austral es uach.cl. Puede encontrar nuevos subdominios dentro de un dominio, como el de la Facultad de Ciencias (fci). Finalmente, un nodo de ese departamento llamado elei tendrá un nombre completo (conocido como totalmente cualificado) tal como elei.fci.uach.cl Este nombre totalmente cualificado también se conoce por las siglas FQDN.

Dependiendiendo de su localización en la jerarquía, un dominio puede ser de primer nivel (top-level), segundo nivel o tercer nivel. Se pueden añadir todos los niveles que queramos, pero no son habituales. Los que siguen son los dominios de primer nivel que veremos con frecuencia:

Tabla 6: Descripción de Algunos Dominios

Dominio	Descripción
Edu	Instituciones universitarias, casi todas norteamericanas.
Com	Organizaciones comerciales.
Org	Organizaciones no comerciales. Las redes privadas UUCP suelen estar en este dominio.

Dominio	Descripción
Net	Pasarelas y otras redes administrativas.
Mil	El ejército norteamericano.
Gov	El gobierno norteamericano.
uucp	Dominio para redes UUCP.

Inicialmente los cuatro primeros dominios de la lista anterior pertenecían sólo a los Estados Unidos, sin embargo, los cambios de política posteriores han hecho que estos dominios, llamados de dominios globales primer nivel (gTLD) sean realmente globales.

Fuera de los Estados Unidos, cada país suele tener su propio dominio de primer nivel codificado con las dos letras del país definidas en la tabla ISO-3166., por ejemplo; en Chile se usa el dominio cl; en México se usa mx; en Argentina, ar, etc. Por debajo de cada dominio de primer nivel, cada país organiza los dominios a su manera. Algunos crean a segundo nivel una serie de dominios similares a los gTLD. Por ejemplo, en Argentina encontramos los dominios com.ar para las empresas, y org.ar para las organizaciones sin ánimo de lucro. Otros países, como España, ponen directamente como nombres de segundo nivel las instituciones o empresas que los solicitan. Por ejemplo, tenemos hispalinux.es.

Por supuesto, el hecho de que un nombre esté en uno de estos dominios nacionales, no implica que la máquina esté realmente en ese país; significa simplemente que ha sido registrada en el NIC de ese país. Un fabricante sueco puede tener oficinas en Australia y tener sus computadores de allá registrados en el dominio se.

La organización del espacio de nombres en una jerarquía de nombres de dominio sirve para resolver fácilmente el problema de la unicidad de los nombres; además muchos nombres completamente cualificados son fáciles de recordar. Bajo esta premisa es conveniente dividir un dominio con gran número de máquinas en subdominios.

El sistema DNS hace más cosas. Permite delegar la *autoridad* de un subdominio a sus administradores. Por ejemplo, los responsables del servidor elei pueden crear un subdominio para cada departamento, y delegar su control a éstos. Así, cada departamento puede definir libremente todos los nodos que quiera dentro de su subdominio e incluso crear nuevos subdominios y delegarlos.

Para esto, el espacio de nombres se divide en *zonas*, cada una asignada a un dominio. Hay que ver la diferencia entre *zona* y *dominio*: por ejemplo, el dominio uach.cl incluye todas las

máquinas y subdominios de éste. Mientras que la zona uach.cl solo incluye las máquinas del dominio, no los subdominios delegados. Es decir, los nodos del subdominio fci.uach.cl pertenecen a una zona diferente.

19.1 Búsquedas con DNS

Ahora verá la parte más ingeniosa del DNS. La idea es que si quiere buscar la dirección IP del sistema elei, DNS pensará, “Preguntemos a la gente que lo maneja, y nos lo dirá.” De hecho, el DNS es como una gigantesca base de datos distribuida. Está realizada a través de los llamados servidores de nombres, que proporcionan la información de uno o varios dominios. Para cada zona, debe haber dos o más servidores de nombres capaces de responder por ella. Para obtener la dirección IP de elei, todo lo que necesitamos es contactar con el servidor de nombres de la zona uach.cl y obtendremos los datos solicitados.

Ahora, ¿cómo se llega al servidor de nombres de la Universidad Austral? Cuando una aplicación quiera buscar la información de elei, contactará con un servidor de nombres local, quien lleva a cabo una secuencia de peticiones. En primer lugar pregunta al servidor de nombres raíz, preguntando por elei.fci.uach.cl. El servidor raíz reconoce que el nombre no pertenece a ninguna de sus zonas *de autoridad* pero sí sabe qué hacer con la zona *cl*. Esto es, devuelve a nuestro servidor más información sobre los servidores de nombres que pueden servir la zona cl. Ahora, el servidor preguntará por este nombre a uno de esos servidores. Ellos enviarán a uno que tenga información autorizada del dominio uach.cl. Luego, el servidor interrogará a éste y finalmente obtendrá la dirección de elei.

Aparentemente la búsqueda de una dirección IP supone mucho tráfico, sin embargo es minúsculo si lo comparamos con la consulta de un gigantesco fichero HOSTS.TXT. Aun así hay técnicas para mejorar el rendimiento.

Para acelerar futuras peticiones de nombres, el servidor almacena la información obtenida en la búsqueda anterior en su *cache* local. Así, la próxima vez que busque algún nodo de uach.cl, ya no tendrá que ir a los servidores raíz o los de la zona cl. Por supuesto que el servidor de nombres no almacenará para siempre la información en la cache; la limpiará cada cierto tiempo. El tiempo de vida se llama *TTL* (del inglés *time to live*). En cada zona DNS el administrador asigna un valor de TTL.

19.2 Tipos de servidores de nombres

Los servidores de nombres que mantienen oficialmente la información de una zona se conocen como *autorizados* de la zona, y a veces se conocen como *servidores principales o maestros*. Cualquier petición de nodos de esa zona irá a parar a uno de estos servidores principales.

Los servidores principales deben estar bien sincronizados. Es decir, uno de ellos será llamado *primario*, que carga su información de un fichero, y los demás servidores *secundarios*, obtienen su información pidiéndosela periódicamente al primario.

El objetivo de tener varios servidores es distribuir la carga y dar cierta tolerancia a fallos. Cuando uno de los servidores principales falla, todas las peticiones son repartidas a los demás. Por supuesto, este esquema no protege de fallas del servidor que produzcan errores en todas las peticiones DNS, como podrían ser errores del software.

También puede instalar un servidor de nombres que no es maestro de ninguna zona. Esto es útil, para dar servicio de nombres a una red local aprovechando sus características de ahorro de ancho de banda gracias a su cache. Estos servidores se conocen como de *solo-cache*.

19.3 La base de datos DNS

Se ha visto que el DNS no solo sabe de direcciones IP de máquinas, también almacena otras informaciones.

Cada unidad de información del DNS se llama *Registro de Recurso* (RR). Cada registro tiene un tipo asociado que describe el dato que contiene, y una clase que especifica el tipo de red al que se aplica. Esto último se adapta a diferentes esquemas de dirección, como direcciones IP (la clase IN), direcciones Hesiod (utilizadas por el sistema Kerberos del MIT) y algunas más. El RR típico es el registro A, que asocia un nombre completamente cualificado con una dirección IP.

Un nodo puede ser conocido por más de un nombre. Por ejemplo, podemos tener un servidor que proporciona tanto servicio FTP como WWW, y tendrá dos nombres: ftp.maquinas.org y www.maquinas.org. Sin embargo, uno de estos nombres debe ser

identificado como oficial o *canónico*. La diferencia es que el canónico es el único registro A que debe existir apuntando a esa dirección IP, mientras que el resto de los nombres deben ser alias (registros CNAME), que apuntan al nombre canónico.

19.4 Resolución inversa

La operación más habitual con el DNS es obtener la dirección IP correspondiente a un nombre de nodo. Sin embargo, a veces se desea hacer la operación opuesta: encontrar el nombre a partir de la dirección IP. Esto se conoce como *resolución inversa*, y la usan diversas aplicaciones para comprobar la identidad del cliente. Cuando se utiliza el fichero hosts, la resolución se realiza mediante una búsqueda simple en el fichero. Con el DNS, una búsqueda exhaustiva en el espacio de nombres carece de sentido. En su lugar, existe un dominio especial, el in-addr.arpa, que contiene las direcciones IP de todos los sistemas en una notación de puntos invertida. Por ejemplo, a la dirección 1.2.3.4 le corresponde el nombre 4.3.2.1.in-addr.arpa. El registro de recurso (RR) que define esto se llama registro PTR.

Cuando se crea una zona de autoridad, ello suele significar que sus administradores tienen control total sobre cómo se asignan los nombres a las direcciones. Puesto que normalmente tienen bajo su control una o más redes o subredes IP, se da una situación de mapeo uno-a-varios entre zonas DNS y redes IP. El departamento de Electrónica, por ejemplo, comprende las subredes 149.76.8.0, 149.76.12.0 y 149.76.14.0. En consecuencia, deben crearse nuevas zonas en el dominio in-addr.arpa para la zona de Electrónica, delegándose a ésta las siguientes: 8.76.149.in-addr.arpa, 12.76.149.in-addr.arpa, y 14.76.149.in-addr.arpa. De otro modo, cada vez que instalásemos un nuevo nodo en el laboratorio de computación, habría que contactar con el que gestiona la red padre para que actualizase su fichero de zona in-addr.arpa.

Las zonas de in-addr.arpa sólo pueden ser creadas por súper conjuntos de redes IP. Hay una restricción más severa: las máscaras de estas redes deben contener los octetos completos. Es decir, podemos crear una zona para una red con máscara 255.255.255.0 pero no para una del tipo 255.255.255.128. El motivo es que para especificar la red delegada 149.76.4.0 tenemos el dominio 4.76.149.in-addr.arpa, pero para la red 149.76.4.128 no tenemos forma de nombrar el dominio in-addr correspondiente.

19.5 Ejecución de named

Named es el servidor DNS en casi todas las máquinas Linux. Es un programa desarrollado originalmente para BSD. La versión 4 se ha usado mucho tiempo y venía con cualquier distribución Linux. La nueva edición, la versión 8, se ha introducido después y supone grandes cambios desde la anterior. Tiene muchas características nuevas, como el soporte de actualización dinámica del DNS, notificaciones de cambios, mejoras importantes de rendimiento y una nueva sintaxis de fichero de configuración. Para más detalle puede referirse a la documentación que viene con el código fuente.

Named suele iniciarse al arrancar la máquina, y ejecutarse hasta que se apaga. Las versiones anteriores de BIND hasta la 8 obtienen la información que necesitan de un fichero llamado /etc/named.boot. Las nuevas versiones usan el fichero /etc/named.conf. Además, hay que configurar los ficheros de zona.

Para ejecutar named, solo tiene que teclear:

```
# /usr/sbin/named
```

El programa named se iniciará y leerá el fichero named.boot y los ficheros de zona que se especifiquen en él. Su número de proceso será anotado en ASCII en el fichero /var/run/named.pid, recibirá ficheros de zona de los servidores principales si es necesario y comenzará a escuchar las peticiones de DNS por el puerto 53.

19.6 El fichero named.boot

El fichero de configuración para los BIND anteriores a la 8 tenía una estructura muy simple. En la versión 8 el fichero se llama /etc/named.conf y es totalmente distinto. Se analizarán ambas versiones y verá las diferencias, y cómo convertir del formato antiguo al nuevo.

El fichero named.boot suele ser muy pequeño y contiene punteros a ficheros con información de zonas y a otros servidores de nombres. Los comentarios en este fichero comienzan con un punto y coma y se extienden hasta el siguiente fin de línea. Antes de que se

vea con más detalle el formato de este fichero, observe el ejemplo para la máquina llaima dado a continuación.

```
;
; Fichero /etc/named.boot para llaima.telsur.cl
;
directory     /var/named
;           domain                          file
;-------------------------------------------------------------------
cache       .                               named.ca
primary     telsur.cl                       named.hosts
primary     0.0.127.in-addr.arpa            named.local
primary     16.172.in-addr.arpa             named.rev
```

La palabra directory indica a named el directorio donde están los demás ficheros de configuración (los ficheros de zona). Los comandos cache y primary sirven para cargar información en \prog {named}. Esta información se obtiene de los ficheros especificados en el segundo argumento. Contienen representaciones textuales de los registros DNS, que se verán a continuación.

En este ejemplo, se configura named como el servidor de nombres principal para tres dominios: los que se indican con el comando primary. La primera línea dice que named actúe como servidor principal para telsur.cl, tomando la información de zona del fichero named.hosts.

La entrada iniciada con la palabra cache es muy especial y debe estar presente en casi todas las máquinas que ejecuten un servidor de nombres. Su función es doble: indica a named que active su *cache*, y también que cargue la información de los servidores raíz del fichero indicado (en este caso, named.ca).

A continuación se presenta una lista de las opciones más importantes que puede poner en el fichero named.boot:

directory

Especifica un directorio donde estén los ficheros de zona. Pueden ponerse varios directorios repitiendo el comando directory. De acuerdo con el estándar de sistema de ficheros para Linux, el directorio debería ser /var/named.

primary

Los argumentos que lleva son un nombre de dominio y un nombre de fichero, declarando el servidor local primario para el dominio de named. Como servidor primario, named carga la información de zona del fichero dado.

Normalmente, siempre habrá por lo menos un comando primary en cada fichero named.boot, para traducción inversa del IP 127.0.0.1, que es el interface de bucle o «loopback».

secondary

Esta sentencia tiene como parámetros un nombre de dominio, una lista de direcciones y un nombre de fichero. Declara el servidor local como servidor maestro secundario para el dominio indicado.

Un servidor secundario mantiene también información «autorizada» como el primario, pero en lugar de obtenerla de un fichero, la intenta obtener de un servidor primario. Debe proporcionarse al menos una dirección IP de servidor primario en la lista de direcciones. El servidor local irá contactando con cada uno de ellos hasta que transfiera con éxito la base datos de zona, que será almacenada en el fichero de respaldo -copia de seguridad o backup- dado en el tercer argumento del comando. Si ninguno de los servidores primarios responde, se obtendrá la información de zona del fichero de respaldo.

Named intentará entonces refrescar los datos almacenados regularmente. Esto se explica después cuando se vean las entradas SOA de los ficheros.

cache

Tiene como argumentos un dominio y un nombre de fichero. Contiene la lista de servidores de nombres raíz. Sólo se reconocerán registros NS y A. El argumento *domain* es normalmente el nombre del dominio raíz (.). Esta información es fundamental: si el comando cache no existiera, named no haría una *cache* local. Esto degradaría de forma importante el rendimiento e incrementaría la carga de la red si los nombres que se buscan no están en la red local. Además, named tampoco será capaz de contactar con cualquier servidor de nombres raíz, y por ello, no podrá resolver ninguna dirección excepto aquellas para las que esté autorizado. Una excepción a esta regla, ocurre cuando se usan servidores redirigidos (con la opción forwarders explicada a continuación).

forwarders

Esta opción lleva una lista de direcciones como argumento. Las direcciones IP en la lista especifican servidores de nombres a los que named puede preguntar si falla una traducción de un nombre mediante su *cache* local. Se intenta preguntar a todos en orden hasta que uno de ellos responda.

slave

Esta opción hace que el servidor sea *esclavo*. Esto significa que nunca realizará consultas recursivas, sino que las redirigirá a los servidores especificados con forwarders. Hay dos opciones adicionales que se van a discutir: sortlist y domain. Además, hay dos directivas que pueden aparecer en los ficheros de zona. Son $INCLUDE y $ORIGIN, que tampoco van a ser tratadas, ya que raramente se utilizan.

19.7 El fichero named.conf de BIND 8

En la versión 8 de BIND se han incluido nuevas características, lo cual ha requerido una nueva sintaxis del fichero de configuración principal. El fichero named.boot ha sido reemplazado por otro, de nombre named.conf, que tiene una sintaxis similar a la del programa gated y recuerda a la del lenguaje C.

La nueva sintaxis es más compleja, afortunadamente se dispone de una utilidad para convertir automáticamente los ficheros named.boot de sintaxis antigua. Esta utilidad es un script de PERL llamado named-bootconf.pl, cuyo código fuente se encuentra en el BIND 8; y lee un fichero en sintaxis antigua, devolviendo por su salida estándar el fichero en sintaxis nueva. Naturalmente, para utilizarlo es necesario tener correctamente instalado el intérprete de lenguaje PERL.

Al script lo invoca, por ejemplo, así:

```
# cd /etc
# named-bootconf.pl <named.boot >named.conf
```

El script produce entonces un fichero similar al que se muestra en el próximo ejemplo, donde se han eliminado algunos comentarios que produce adicionalmente el script.

```
//
   // /etc/named.boot para vlager.vbrew.com
   options {
         directory "/var/named";
   };
      zone "." {
         type hint;
         file "named.ca";
   };

   zone "vbrew.com" {
         type master;
         file "named.hosts";
   };

   zone "0.0.127.in-addr.arpa" {
         type master;
         file "named.local";
   };

   zone "16.172.in-addr.arpa" {
         type master;
         file "named.rev";
   };
```

Si observa el ejemplo, verá que cada línea de named.boot ha sido convertida a un bloque en estilo C, encerrado entre llaves (signos { y }). Los comentarios del fichero se escriben ahora en notación similar a C++, es decir, dos barras (signo //).

La sentencia directory va ahora dentro del bloque options, junto a otras posibles opciones globales de configuración. Las sentencias cache y primary se convierten en bloques de zona, con sentencias type específicas, de valor hint y master, respectivamente. Los ficheros de zona no necesitan modificarse, ya que su sintaxis sigue siendo la de antes.

La nueva sintaxis de configuración se ha pensado para poder incluir muchas más opciones de configuración, en las que no se van a detallar. Si desea conocerlas, la mejor fuente de información es la que viene con el paquete de fuentes de BIND 8.

19.8 Ficheros de base de datos DNS

Los ficheros incluidos con named, como named.hosts, siempre tienen un dominio asociado a ellos llamado *origen*. Este es el nombre de dominio especificado con los comandos cache y primary. En un fichero maestro, se pueden especificar nombres de máquinas y dominios relativos a este dominio. Un nombre dado en un fichero de configuración se considera \emph{absoluto} si termina con un punto. En caso contrario se considera relativo al origen. Al origen en sí mismo nos podemos referir con «@».

Todos los datos en un fichero principal se dividen en *registros de recursos* o RRs. Son la unidad de información del DNS. Cada RR tiene un tipo. Los registros de tipo A, por ejemplo, asocian un nombre a una dirección IP, como se comentó al comienzo de la sección. Los registros de tipo CNAME asocian un alias de una máquina con su nombre oficial.

19.9 Cómo verificar la configuración

nslookup es una buena utilidad para comprobar el funcionamiento de un servidor de nombres. Se puede usar interactivamente o pasándole la pregunta por la línea de comandos. En este último caso puede invocar el comando así:

```
$ nslookup
nombre-de-host
```

nslookup envía sus peticiones al servidor citado en resolv.conf. Si este fichero tiene más de un servidor, nslookup eligirá uno al azar.

El modo interactivo es mucho más interesante. No solo sirve para buscar la IP de un nodo, sino que también puede interrogar acerca de cualquier tipo de registro DNS y transferir toda la información de una zona si así lo quiere.

Si se invoca sin argumentos, nslookup muestra el nombre del servidor elegido y entra en modo interactivo. En el prompt > debe escribir cualquier nombre de dominio. Al principio preguntará solo por registros A, es decir, obtención de la IP asociada.
Puede elegir un tipo de registro diferente con el comando:

```
> set type=tipo
```

donde *tipo* es uno de los tipos de RR descritos antes, o ANY.
Vea una posible sesión de nslookup:

```
$ nslookup
   Default Server:  tao.linux.org.au
   Address:  203.41.101.121

   > metalab.unc.edu
   Server:  tao.linux.org.au
   Address:  203.41.101.121

   Name:    metalab.unc.edu
   Address:  152.2.254.81
   >
```

La salida muestra el servidor DNS interrogado y el resultado obtenido. Si pregunta por algo que no tiene IP asociada pero sí otros registros de otra clase, el programa nos devolverá una advertencia del tipo No type A records found. Sin embargo, puede usar el citado comando set type para buscar registros de otras clases. Por ejemplo, el registro SOA de un dominio puede ser pedido así:

```
> unc.edu
   Server:  tao.linux.org.au
   Address:  203.41.101.121

   *** No address (A) records available for unc.edu
```

```
> set type=SOA
> unc.edu
Server:  tao.linux.org.au
Address:  203.41.101.121

unc.edu
        origin = ns.unc.edu
        mail addr = host-reg.ns.unc.edu
        serial = 1998111011
        refresh = 14400 (4H)
        retry   = 3600 (1H)
        expire  = 1209600 (2W)
        minimum ttl = 86400 (1D)
unc.edu name server = ns2.unc.edu
unc.edu name server = ncnoc.ncren.net
unc.edu name server = ns.unc.edu
ns2.unc.edu     internet address = 152.2.253.100
ncnoc.ncren.net internet address = 192.101.21.1
ncnoc.ncren.net internet address = 128.109.193.1
ns.unc.edu      internet address = 152.2.21.1
```

De manera parecida, para preguntar por registros MX hará:

```
> set type=MX
> unc.edu
Server:  tao.linux.org.au
Address:  203.41.101.121

unc.edu preference = 0, mail exchanger = conga.oit.unc.edu
unc.edu preference = 10, mail exchanger = imsety.oit.unc.edu
unc.edu name server = ns.unc.edu
unc.edu name server = ns2.unc.edu
unc.edu name server = ncnoc.ncren.net
```

```
conga.oit.unc.edu       internet address = 152.2.22.21
imsety.oit.unc.edu      internet address = 152.2.21.99
ns.unc.edu      internet address = 152.2.21.1
ns2.unc.edu     internet address = 152.2.253.100
ncnoc.ncren.net internet address = 192.101.21.1
ncnoc.ncren.net internet address = 128.109.193.1
```

Con el tipo ANY obtendrá todos los registros existentes asociados al nombre dado. Una aplicación práctica de nslookup, para depurar un servidor, es obtener la lista de servidores raíz. Para ello no hay más que pedir los NS del registro raíz (.):

```
> set type=NS
Server:  tao.linux.org.au
Address:  203.41.101.121

Non-authoritative answer:
(root)  name server = A.ROOT-SERVERS.NET
(root)  name server = H.ROOT-SERVERS.NET
(root)  name server = B.ROOT-SERVERS.NET
(root)  name server = C.ROOT-SERVERS.NET
(root)  name server = D.ROOT-SERVERS.NET
(root)  name server = E.ROOT-SERVERS.NET
(root)  name server = I.ROOT-SERVERS.NET
(root)  name server = F.ROOT-SERVERS.NET
(root)  name server = G.ROOT-SERVERS.NET
(root)  name server = J.ROOT-SERVERS.NET
(root)  name server = K.ROOT-SERVERS.NET
(root)  name server = L.ROOT-SERVERS.NET
(root)  name server = M.ROOT-SERVERS.NET

Authoritative answers can be found from:
A.ROOT-SERVERS.NET      internet address = 198.41.0.4
H.ROOT-SERVERS.NET      internet address = 128.63.2.53
```

```
B.ROOT-SERVERS.NET      internet address = 128.9.0.107
C.ROOT-SERVERS.NET      internet address = 192.33.4.12
D.ROOT-SERVERS.NET      internet address = 128.8.10.90
E.ROOT-SERVERS.NET      internet address = 192.203.230.10
I.ROOT-SERVERS.NET      internet address = 192.36.148.17
F.ROOT-SERVERS.NET      internet address = 192.5.5.241
G.ROOT-SERVERS.NET      internet address = 192.112.36.4
J.ROOT-SERVERS.NET      internet address = 198.41.0.10
K.ROOT-SERVERS.NET      internet address = 193.0.14.129
L.ROOT-SERVERS.NET      internet address = 198.32.64.12
M.ROOT-SERVERS.NET      internet address = 202.12.27.33
```

Para ver el conjunto completo de comandos, puede usar help dentro de nslookup.

19.10 Otras Utilidades Interesantes

Tras la puesta en marcha del servidor, deberá probarla. Existen utilidades para esto. Una de ellas es dnswalk, que está basada en Perl. La segunda es nslint. Ambas recorren la base de datos DNS buscando errores habituales y verificando que la información que encuentran es consistente. Otras dos utilidades interesantes son host y dig, que vienen con el paquete BIND y pueden usarse para una inspección manual del las bases de datos.

Estas utilidades suelen venir empaquetadas. dnswalk y nslint están disponibles en fuentes en http://www.visi.com/~barr/dnswalk/ y ftp://ftp.ee.lbl.gov/nslint.tar.Z. En cuanto a host y dig, el código fuente se encuentra en ftp://ftp.nikhef.nl/pub/network/ y ftp://ftp.is.co.za/networking/ip/dns/dig/.

20. Correo Electrónico

Uno de los usos más comunes de las redes informáticas desde sus orígenes ha sido el correo electrónico. Empezó siendo un simple servicio que copiaba un fichero de una máquina a otra, y lo añadía al fichero *mailbox* (buzón de correo) del destinatario. Básicamente, en esto sigue consistiendo el e-mail (correo electrónico), aunque el crecimiento continuo de la red y, consiguientemente, el aumento de la complejidad de encaminado, ha hecho necesario un esquema más elaborado.

Se han diseñado varios estándares de intercambio de correo. Los nodos conectados a Internet cumplen uno recopilado en el RFC 822, complementado en algunos RFCs que describen un método independiente de la máquina para transferir casi *cualquier contenido*, incluso gráficos, archivos de sonido y conjuntos de caracteres especiales. El CCITT definió otro estándar, el X.400. Aún se usa en ambientes de grandes corporaciones y gobiernos, pero está siendo retirado progresivamente.

Hay ya una gran cantidad de programas de transporte de correo para sistemas Unix. Uno de los más conocidos es el sendmail, desarrollado por Eric Allman en la Universidad de California, en Berkeley. Eric Allman ofrece ahora sendmail como un producto comercial, pero el programa sigue siendo software libre. Sendmail se ofrece como el agente de correo estándar en algunas distribuciones de Linux. Se verá la configuración de sendmail mas adelante.

Linux también usa Exim, escrito por Philip Hazel de la Universidad de Cambridge. Comparado con sendmail, Exim es bastante joven, para la gran mayoría de los sitios con requerimientos de correo electrónico, sus capacidades son muy parecidas. Ambos admiten un conjunto de ficheros de configuración que deben ser adaptados a cada caso particular. Aparte de la información que se necesita para hacer funcionar el subsistema de correo (como puede ser el nombre del computador local), hay muchos más parámetros que deben ajustarse. El fichero principal de configuración de sendmail es muy difícil de entender al principio. Los ficheros de configuración de Exim están más estructurados y son más fáciles de entender que los del sendmail, pero no ofrecen soporte directo para UUCP y manejan sólo direcciones de dominio. Hoy esto no es una gran limitación como lo era anteriormente; en cualquier caso, para la mayoría de los sitios, el trabajo requerido en configurar ambos es aproximadamente el mismo.

A continuación se tratará sobre qué es el correo electrónico y que factores tiene que abordar un administrador del sistema. La información que se suministra debe bastar para poner en marcha pequeños nodos, pero hay muchas más opciones.

Para mas información sobre temas específicos de correo electrónico sobre Linux, debe consultar el 'Electronic Mail HOWTO' de Guylhem Aznar, que aparece en comp.os.linux.answers con regularidad. Las distribuciones fuente de elm, Exim, y sendmail contienen también una documentación muy extensa que debe solucionar la mayoría de sus dudas sobre instalación y puesta a punto. Si busca información sobre correo electrónico en general, hay varios RFCs que tratan específicamente este tema.

20.1 ¿Cómo se reparte el correo?

Generalmente, escribe su correo usando un interface de correo como mail o mailx u otros más sofisticados como mutt, tkrat, o pine. Estos programas se denominan *agentes de usuario de correo* (mail user agents), o MUAs para abreviar. Si envía un mensaje de correo, el programa interface en la mayoría de los casos se lo pasará a otro programa para que lo transmita. Este programa se denomina *agente de transporte de correo* (mail transport agent), o MTA. En la mayoría de los sistemas se usa el mismo MTA tanto para el reparto local como remoto y, normalmente se invoca como /usr/sbin/sendmail o, en algunos sistemas, /usr/lib/sendmail. En sistemas UUCP no es raro ver que el correo se reparte por dos programas distintos: rmail para el envío remoto de correo, y lmail para el reparto local.

Un envío local de correo es, por supuesto, algo más que añadir el mensaje al buzón del destinatario. Usualmente el MTA local entenderá como usar alias (definir direcciones locales de destinatarios que dirigen a otras direcciones) y como usar redirecciones (dirigir el correo de un usuario a otra dirección). Además, los mensajes que no pudieron ser enviados deben ser normalmente *devueltos* (bounced) al remitente junto con algún mensaje de error.

Para envíos lejanos, el software de transporte usado depende del tipo de enlace. Si el correo debe enviarse a través de una red que usa TCP/IP, se usará normalmente *Protocolo Simple de Trasferencia de Correo* (SMTP), que se define en el RFC 821. SMTP se diseñó para repartir correo directamente en la máquina de un destinatario, negociando la transferencia del mensaje con el demonio SMTP del lado remoto. Hoy es práctica común de las organizaciones establecer máquinas especiales que aceptan todo el correo para destinatarios de la

organización, y estos nodos se encargan de controlar el reparto apropiado a los destinos adecuados.

En redes tipo UUCP, el correo no suele ser enviado directamente, sino que es redirigido hasta su destino a través de un conjunto de máquinas intermedias. Para enviar un mensaje a través de un enlace UUCP, el MTA remitente ejecutará usualmente rmail en la máquina intermedia usando uux, y suministrándole el mensaje en la entrada estándar.

Desde que se llama a uux separadamente para cada mensaje, puede producirse una carga considerable en un nodo procesador de correo, además de inundar las colas UUCP con cientos de pequeños mensajes que ocupan una cantidad de disco desproporcionada. Por esto algunos MTAs permiten recopilar varios mensajes de un sistema remoto en un solo lote. El fichero de lotes contiene los comandos SMTP que el nodo local ejecutaría normalmente si usara una conexión SMTP directa. A esto se le llama BSMTP, o *batched* SMTP (SMTP por lotes). El lote es suministrado a los programas rsmtp o bsmtp en el sistema remoto, que procesará la entrada como si se hubiera dado una conexión SMTP normal.

20.2 Direcciones de correo electrónico

Las direcciones de correo electrónico constan de dos partes: la primera es el nombre de un *dominio de correo* encargada de traducir la información, o bien al anfitrión del receptor o a cualquiera que acepte correo de su parte. La segunda es la identificación exclusiva del usuario que puede ser tanto el nombre que le permite el acceso al sistema, como el nombre del usuario en formato “nombre.apellido”, o un alias que se transmitirá a un usuario o a la lista de usuarios. Otros formatos de dirección de correo como el X.400 utilizan otra serie de “atributos” que sirven para localizar al sistema anfitrión del receptor en el directorio del servidor X.400.

La interpretación de las direcciones de correo electrónico depende en gran medida del tipo de red de la que usted disponga. Ahora verá cómo los protocolos TCP/IP y UUCP interpretan las direcciones de correo electrónico.

20.3 RFC-822

Los sitios de Internet están ligados al estándar RFC-822, que requiere la conocida notación usuario@anfitrión.dominio para el cual anfitrión.dominio es el nombre de dominio más adecuado para el anfitrión. El símbolo que separa ambas partes recibe el nombre de “arroba” en inglés “at.” Esta nomenclatura no especifica la ruta al sistema anfitrión. Otros mecanismos que se tratarán en breve son los encargados del enrutamiento de los mensajes.

EL estándar RFC-822 no es exclusivo del correo, ya que se ha extendido a otros servicios como los grupos de noticias.

En el entorno original UUCP la forma corriente era ruta!anfitrión!usuario, para el cual la ruta indicaba una secuencia de anfitriones por los que el mensaje tenía que pasar antes de llegar a su destino host. Este modelo recibe el nombre de notación *bang path* , porque en inglés coloquial, la exclamación se conoce con el nombre “bang.” Actualmente, muchas de las redes basadas en UUCP han adoptado el formato RFC-822 y aceptan las direcciones de correo basadas en el dominio.

Hay redes que usan otros sistemas de dirección. Por ejemplo, las basadas en DECnet, usan los dos puntos (:) como elemento separador de sus partes, resultando la dirección anfitrión::usuario. El estándar X 400 utiliza un estilo totalmente diferente, describiendo al receptor por medio de pares de atributos como país y organización a la que éste pertenece.

En último lugar, está FidoNet, en donde cada usuario se identifica con un código como 2:320/204.9 , que consiste en cuatro números que indican la zona donde se encuentra (el 2 es para Europa), la red (el 320 se refiere a Paris y Banlieue), el nodo (distribuidor local), y el punto de conexión (el computador del usuario). Se puede trabajar con direcciones Fidonet en RFC-Thomas.Quinot@p9.f204.n320.z2.fidonet.org (bastante difícil de recordar). Teniendo varias formas de notación, la pregunta obvia, es cómo combinar distintos formatos de correo electrónico. Cuando a un conjunto de sistemas le sumamos gente inteligente, lo normal es que se intente buscar maneras para poder conectarse entre sí y trabajar en red. Por consiguiente, existen distintas pasarelas de correo que vinculan dos sistemas diferentes, de forma que el correo pueda ser transmitido de uno a otro. El problema más crítico a la hora de interconectar dos sistemas es el referente a las direcciones de correo.

Imagine que quiere trabajar con la notación bang-path de UUCP y RFC-822. No son dos formatos fáciles de combinar. Suponga que tiene la siguiente dirección dominioA!usuario@dominioB. No está claro si el símbolo @ tiene prioridad sobre la ruta, o

viceversa: entonces, ¿tendría que mandar el mensaje adominioB, que lo enviaría a dominioA!usuario, o por el contrario debería hacerlo a dominioA, que lo dirigiría a usuario@dominioB ?

Aquellas direcciones formadas por distintos proveedores de correo se llaman *hybrid addresses*. El tipo más común, que es el que se acaba de ilustrar, normalmente se resuelve dando prioridad al símbolo @ sobre la ruta. Esto significaría enviar primero el mensaje a dominioB en dominioA!usuario@dominioB. Sin embargo, hay una manera de especificar las ruta en RFC-822: <@dominioA,@dominioB:usuario@dominioC > indica la dirección del usuario en el dominioC, donde llega al dominioC pasando por dominioA y dominioB (en ese orden). Este tipo de dirección se llama con frecuencia dirección *encaminada desde la fuente* Tampoco es bueno basarse exclusivamente en este sistema, ya que un posterior repaso al estándar RFC en su apartado de la descripción del enrutamiento del correo, recomienda que se intente enviar el correo directamente a su destino remoto en lugar de hacerlo por medio de la fuente.

Existe además el % proveedor de correo usuario %dominioB@dominioA que lo envía primero al dominioA, y transforma el símbolo de porcentaje más indicado (que en este caso es el único) a un símbolo arroba@ sign. La dirección en este caso es usuario@dominioB, y el mensajero dirige su mensaje a dominioB, el cual lo envía al usuario. A este tipo de dirección se le suele llamar “Ye Olde ARPAnet Kludge,”.

El uso de estos tipos distintos de direccionamiento puede tener sus repercusiones, las cuales verá en las secciones siguientes. En un entorno RFC-822 no se deberá usar otra dirección que no sea una absoluta como usuario@anfitrión.dominio.

20.4 ¿Cómo funciona el enrutamiento del correo?

Se conoce como *elección de rutas* (routing) el proceso de dirigir un mensaje al sistema anfitrión del receptor. Aparte de localizar una ruta desde el sitio del emisor hasta el de destino, la elección de rutas implica la detección de errores e incluso la optimización de la velocidad y el coste.

Hay una gran diferencia entre la elección de rutas por parte de un sitio UUCP y un sitio de Internet. En Internet, la función principal a la hora de dirigir los datos al anfitrión del receptor (cuando el protocolo IP lo conoce), la realiza la capa de red IP, mientras que en el entorno

UUCP, la ruta tiene que ser provista por el usuario o generado por el agente de transmisión de correo.

La configuración del sistema anfitrión del destinatario determina si se está trabajando con un determinado sistema de localización de la ruta de correo en Internet. La opción por defecto es transmitir el mensaje a su destino determinando primero el anfitrión al que debe ser enviado, y mandándolo allí directamente. La mayoría de los sitios de Internet buscan dirigir todo el correo entrante a un servidor de correo con alta disponibilidad capaz de manejar todo el tráfico y distribuirlo localmente. Para dar a conocer este servicio, el sitio publica el llamado registro MX para su dominio local en su base de datos DNS. MX (Mail Exchanger) significa *Mail Exchanger* y básicamente indica que el anfitrión del servidor es capaz de convertirse en emisor de correo para todas las direcciones del dominio. Los registros MX también pueden manejar el tráfico de anfitriones que no están conectados a la red, como UCCP o FidoNet, que necesitan una pasarela de correo.

Los registros MX siempre tienen asignada un *nivel de preferencia*. Si son muchos los proveedores de correo existentes (MX) para un anfitrión, el agente de transporte de correo tratará de enviar el mensaje al proveedor cuya preferencia sea la menor. Sólo si esta operación falla, lo enviará a un anfitrión de mayor índice de preferencia. Si el anfitrión local es el proveedor de correo para la dirección de destino, puede enviar los mensajes a un anfitrión de menor preferencia que él mismo; esta es una manera segura de evitar los bucles en el correo. Si no hay ningún registro MX para un determinado dominio, o no es disponible, el agente de transporte de correo puede comprobar si la dirección IP del dominio está asociada a él, y así intentar mandarlo directamente a ese anfitrión.

Suponga que hay una organización dada, por ejemplo Foobar, Inc., que quiere que todo su correo lo controle el servidor de correo de su computador. Por ello llevará registros MX como el que se muestra a continuación, en la base de datos DNS:

```
green.foobar.com.       IN  MX    5    mailhub.foobar.com.
```

Esto da a conocer a mailhub.foobar.com como proveedor de correo para green.foobar.comcon un nivel de preferencia de 5. Un anfitrión que pretenda enviar un mensaje ajoe@green.foobar.com revisa la base de datos DNS y busca el MX en el distribuidor de correo. Si no hay ningún MX con una preferencia menor a 5, el mensaje se envía al distribuidor de correo que lo entrega a green.

Ésta es una descripción muy básica de cómo funcionan los registros MX. Para obtener más información sobre la elección de rutas de correo en Internet, puede consultar las siguientes RFC: RFC-821, RFC-974 y RFC-1123.

20.5 Introducción a sendmail

Sendmail es un programa increíblemente potente, y también, para la mayoría de las personas, increíblemente difícil de aprender y comprender. Un programa cuyo manual definitivo de referencia (sendmail, por Bryan Costales y Eric Allman, publicado por O'Reilly), ocupa 1,050 páginas, lo que es suficiente para espantar a cualquiera. Información sobre referencias a sendmail se pueden encontrar en la bibliografía, al final de esta Tesis.

Afortunadamente, las nuevas versiones de sendmail son diferentes. Ya no se necesita editar directamente el archivo sendmail.cf; las nuevas versiones proveen de una herramienta de configuración, la cual crea el fichero sendmail.cf por usted basándose en macro-archivos mucho más simples. No se necesitará entender la sintaxis complicada del archivo sendmail.cf; los macro-archivos no lo requieren, en lugar de eso, sólo se necesitará listar ítems, como por ejemplo, el nombre de las características que se desee incluir en vuestra configuración, y especificar algunos de los parámetros que determinan cómo operará esa característica. Para esto se usará una utilidad Unix tradicional llamada m4, la cual toma nuestros macro-archivos de configuración y los combina con los datos obtenidos de las plantillas que contienen la sintaxis actual de sendmail.cf, de forma tal que generará nuestro propio archivo sendmail.cf.

20.6 Instalando Sendmail

El paquete del agente de transporte de correo, sendmail, está incluido en casi todas las distribuciones Linux. En estos casos, la instalación es relativamente simple. Sin embargo, existen algunas razones para instalar sendmail desde el código fuente, especialmente si le importa la seguridad. El programa sendmail es muy complejo y se ha hecho popular, en el correr de los años, por contener fallos que permiten brechas en la seguridad. Uno de los más conocidos ejemplos es el gusano de Internet RTM, el cual sacó provecho de un problema de desbordamiento del búfer en versiones antiguas de sendmail. La mayoría de los abusos que

utilicen desbordamientos del búfer, dependen de todas las copias de sendmail que sean idénticas en las diferentes máquinas, tal como si se tratase de datos almacenados en ubicaciones específicas. Esto es, precisamente lo que ocurre con el programa sendmail que viene ya instalado en una distribución Linux. Compilar sendmail desde el código fuente a mano puede ayudar a reducir este riesgo. Las versiones modernas de sendmail son menos vulnerables, ya que vienen con exámenes extremadamente cuidadosos en cuanto a la seguridad, la cual se ha vuelto una inquietud ampliamente generalizada en la comunidad de Internet.

El código fuente de sendmail está disponible vía FTP anónimo en el servidor ftp.sendmail.org. La compilación es bastante simple, ya que el paquete de las fuentes de sendmail soporta directamente a Linux. Los pasos a seguir para compilar sendmail se resumen en:

```
# cd /usr/local/src
# tar xvfz sendmail.8.9.3.tar.gz
# cd src
# ./Build
```

Para completar la instalación de los archivos binarios resultantes, se necesitará los permisos de root, usando:

```
# cd obj.Linux.2.0.36.i586
# make install
```

Se han instalado ahora los binarios de sendmail en el directorio /usr/sbin. Muchos enlaces simbólicos al ejecutable sendmail también se instalarán en el directorio /usr/bin/.

21. FTP

FTP ("File Transfer Protocol") es el conjunto de programas que se usa en Internet para transferir ficheros entre sistemas. La mayoría de los sistemas UNIX, VMS y MS-DOS de

Internet tienen un programa llamado ftp que se usa para transferir estos ficheros, y si se tiene acceso a Internet, el mejor modo de descargar el software de Linux es usando ftp. Este apartado cubre el uso de ftp a nivel básico, por supuesto, hay muchas más funciones y usos del ftp de los que se mencionan en esta tesis.

Si está usando un sistema MS-DOS, UNIX, o VMS para traer ficheros desde Internet, entonces, ftp será un programa a base de comandos. Sin embargo, existen otras implementaciones de ftp, como la versión Macintosh (llamada Fetch) con un bonito interface guiado por menús, que lo hacen más amigable y cómodo de usar.

Ftp puede usarse tanto para "subir" (enviar) como para "bajar" (recibir) ficheros desde los sitios de Internet. En Internet hay un gran número de sitios FTP de acceso público, máquinas que permiten a cualquiera hacer ftp sobre ellas y descargar el software. Un "archive site" de este tipo es sunsite.unc.edu, que contiene una gran variedad de software de Sun Microsistems, y que actúa como uno de los principales sites de Linux. Además, los FTP archive sites se reflejan (mirror) el software unos a otros, es decir, el software que se sube a un site será automáticamente copiado a un gran número de otros sites. De modo que no hay que sorprenderse si ve exactamente los mismos ficheros en muchos sites diferentes.

21.1 Aprendiendo ftp

Observe en el ejemplo "screens" que figura debajo.

Para arrancar el ftp y conectar con un sitio, usamos simplemente el comando

```
ftp <hostname>
```

donde <hostname> es el nombre del site al que se quiere conectar. Por ejemplo, para conectar con el ya mítico site elei.uach.cl se puede usar el comando

```
ftp elei.uach.cl
```

21.2 Registrándose

Cuando comienza el ftp debería ver algo como

```
Connected to elei.uach.cl.
220 elei.uach.cl FTPD ready at 15 Dec 1992 08:20:42 EDT
Name (elei.uach.cl:fmoraga):
```

En este punto, ftp está pidiendo que le de el nombre de usuario con el que quiere registrarse en elei.uach.cl. El nombre por defecto aquí es fmoraga, que corresponde con el nombre de usuario del sistema. Si no tiene cuenta en elei.uach.cl no se podrá registrar. En cambio, para acceder al software disponible públicamente en un FTP site hay que registrarse como anonymous, y dar la dirección de e-mail Internet (si se tiene) como password. De modo que tendría,

```
Name (elei.uach.cl:fmoraga):   anonymous
331-Guest login ok, send e-mail address as password.
Password:   fmoraga@telsur.net
230- Bienvenido al Servidor de Electrónica.
230- Virtual Pizza Delivery[tm]:   Download pizza in 30 cycles or less
230- or you get it FREE!
ftp>
```

Por supuesto, debería dar una dirección de e-mail, la cual no se reflejará en pantalla mientras la esté escribiendo (ya que técnicamente es un "password" o contraseña). El ftp le debería permitir registrarse y ya estar preparado para descargar el software.

21.3 Moviéndose dentro de los Directorios

Una vez dentro, su prompt es ftp>, el programa ftp está a la espera de comandos. Hay unos pocos comandos básicos que debería conocer. Primero, los comandos

ls <fichero>

y

dir <fichero>

ambos dan un listado de ficheros (donde <fichero> es un argumento opcional que especifica un fichero particular a listar). La diferencia es que ls generalmente produce un listado corto y dir produce un listado más largo (es decir, con más información sobre los tamaños de los ficheros, fechas de modificación, etc.).

Comandos básicos

cd <directorio>

lo moverá al directorio dado (exactamente como el comando cd en sistemas UNIX o MS-DOS). Puede usar también el comando:

cdup

para cambiar al directorio padre.

help <comando>

proporciona ayuda en el citado <comando> ftp (como puede ser ls o cd). Si no se especifica comando, ftp listará todos los comandos disponibles.

Si teclea dir en este punto, verá un listado del directorio inicial en el que se encuentra.

```
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 1337
dr-xr-xr-x      2 root wheel      512 Aug 13 13:55 bin
drwxr-xr-x      2 root      wheel      512 Aug 13 13:58 dev
drwxr-xr-x      2 root      wheel      512 Jan 25 17:35 etc
```

```
drwxr-xr-x          19 root          wheel          1024 Jan 27 21:39 pub
drwxrwx-wx          4 root ftp-admi       1024 Feb   6 22:10 uploads
drwxr-xr-x          3 root           wheel           512 Mar 11   1992 usr

226 Transfer complete.
921 bytes received in 0.24 seconds (3.7 Kbytes/s)
ftp>
```

Cada una de estas entradas es un directorio, no un fichero individual que pueda descargar (especificado por la d de la primera columna del listado). En la mayoría de los "FTP archive sites" (Servidores de FTP), el software públicamente accesible está bajo el directorio /pub,

```
ftp> cd pub
ftp> dir
200 PORT command successful.
150 ASCII data connection for /bin/ls (128.84.181.1,4525) (0 bytes).
total 846
-rw-r--r--      1 root          staff          1433 Jul 12     1988 README
-r--r--r--      1 3807          staff         15586 May 13     1991 US-DOMAIN.TXT.2
-rw-r--r--      1 539           staff         52664 Feb 20     1991 altenergy.avail
-r--r--r--      1 65534         65534         56456 Dec 17     1990 ataxx.tar.Z
-rw-r--r--      1 root          other       2013041 Jul  3     1991 gesyps.tar.Z
-rw-r--r--      1 432           staff         41831 Jan 30     1989 gnexe.arc
-rw-rw-rw-      1 615           staff         50315 Apr 16     1992 linpack.tar.Z
-r--r--r--      1 root          wheel         12168 Dec 25     1990 localtime.o
-rw-r--r--      1 root          staff          7035 Aug 27     1986 manualslist.tblms
drwxr-xr-x      2 2195          staff           512 Mar 10     00:48 mdw
-rw-r--r--      1 root          staff          5593 Jul 19     1988 t.out.h

226 ASCII Transfer complete.
2443 bytes received in 0.35 seconds (6.8 Kbytes/s)
ftp>
```

Aquí puede ver varios ficheros, uno de los cuales se llama README, que debería descargar (la mayoría de FTP sites tienen un fichero README en el directorio /pub).

21.4 Descarga de ficheros

Antes de descargar ficheros, hay algunas cosas que debe tener en cuenta.

a) Activar las marcas de progreso. Las Marcas de Progreso se imprimen en pantalla mientras se están transfiriendo ficheros; permitirán saber como va avanzando la transferencia, y que la conexión no se ha cortado (de modo que no está sentado 20 minutos pensando que aun esta trayéndose un fichero). En general, una marca de progreso aparece como una almohadilla (#), y se imprime una por cada 1024 o 8192 bytes transferidos, dependiendo del sistema.

Para activar la impresión de marcas de progreso, utilice el comando hash.

```
ftp> hash
Hash mark printing on (8192 bytes/hash mark).
ftp>
```

b) Determinar el tipo de fichero que se va a transferir. En lo que concierne al FTP, los ficheros van en dos formatos: binario y texto. La mayoría de los ficheros que puede transferir serán binarios: es decir, programas, ficheros comprimidos, ficheros de datos, etc. Sin embargo, muchos ficheros (tales como READMEs y demás) son ficheros de texto.

¿Por qué es importante el tipo de fichero? Simplemente porque en algunos sistemas (como el MS-DOS), ciertos caracteres de un fichero de texto, tales como retornos de carro, necesitan convertirse de modo que los ficheros puedan ser legibles. Mientras las transferencias en modo binario, no realizan ninguna conversión, el fichero se transfiere byte a byte.

Los comandos bin y ASCII ponen el modo de transferencia en binario y texto respectivamente. Si no está seguro del tipo de archivo, es recomendable siempre usar el modo binario para transferirlos. Si intenta transferir un fichero binario en modo texto, obtendrá un fichero corrupto que no podrá utilizar. (Este es uno de los fallos más corrientes cuando se usa el FTP.) Sin embargo, se puede usar tranquilamente el modo texto para ficheros de texto (aquellos cuyos nombres terminan habitualmente en .txt).

En el ejemplo se traerá el fichero README, que por lo general es un fichero de texto, para lo cual se usa el comando

```
ftp> ASCII
200 Type set to A.
ftp>
```

c) Establecer el directorio local. El directorio local es el directorio del sistema en el que quiere que vayan a parar los ficheros traídos. Mientras el comando cd cambia el directorio remoto (de la máquina remota a la que esta haciendo FTP), el comando lcd cambia el directorio local. Por ejemplo, para cambiar el directorio local a /home/db/mdw/tmp, se usa el siguiente comando

```
ftp> lcd /home/db/mdw/tmp
Local directory now /home/db/mdw/tmp
```

Ahora ya está listo para transferir el fichero con el comando:

```
get <nombre-remoto> <nombre-local>
```

donde <nombre-remoto> es el nombre del fichero de la máquina remota, y <nombre-local> es el nombre que le quiere dar al fichero en la máquina local. El argumento <nombre-local> es opcional; por defecto el nombre fichero local es el mismo que el remoto. Sin embargo, si está trayéndose el fichero README, y ya hay un README en el directorio local, necesita darle un <nombre-local> distinto para que no se sobrescriba el primero.

En nuestro ejemplo, para traer el fichero README, usará simplemente

```
ftp> get README
200 PORT command successful.
150 ASCII data connection for README (128.84.181.1,4527) (1433 bytes).
#
226 ASCII Transfer complete.
local:   README remote:   README
1493 bytes received in 0.03 seconds (49 Kbytes/s)
```

21.5 Envío de ficheros

El comando put se utiliza para enviar archivos a una computadora remota. El comando put envía el archivo local que especifique.

```
ftp> put README
```

Previamente, es necesario tener los permisos correspondientes de escritura en el directorio donde va a enviar el archivo, por lo general en los accesos anonymous no se cuenta con estos atributos, pero si posee una cuenta registrada, como por ejemplo un espacio para la publicación de páginas web puede escribir sobre el directorio local.

21.6 Saliendo de FTP

Para terminar una sesión FTP, solo tiene que usar el comando

```
quit
```

El comando close se puede usar para cerrar la conexión con el FTP site actual; el comando open se puede usar para comenzar una sesión con otro site distinto (sin salir completamente del programa FTP).

```
ftp> close
221 Goodbye.
ftp> quit
```

21.7 Servidores FTP

Tal vez el servicio de FTP es el más sencillo de instalar, por lo general se instala por defecto y su demonio de ejecución es el ftpd,

CAPITULO VI
ENRUTAMIENTO BAJO LINUX

22 Protocolo de Control de Transmisión / Protocolo de Internet (TCP/IP)

Una red de redes consiste en un grupo de redes conectadas que en apariencia actúan como un todo coordinado que proporcionan interconexión universal y permite que grupos individuales utilicen cualquier hardware de red que satisfaga sus necesidades.

El conjunto de Protocolos TCP/IP (Protocolo de Control de Transmisión / Protocolo de Internet) es uno de los más discutidos, ya que de él dependen las redes que actualmente conocemos, incluida la Internet.

Este protocolo surgió en un comienzo como un proyecto gubernamental promovido por el departamento de defensa de los Estados Unidos, ya que por esa época (mediados de los 70) la mayoría de los sistemas de comunicaciones eran de fabricantes distintos, que ejecutaban sistemas operativos distintos con topologías y protocolos de red muy diversas, y a medida que se iba incrementado el volumen de la información no todos los sistemas podían conversar entre ellos y por ende no podían compartir la información.

De esta forma se le pidió a la agencia de Investigación de Proyectos Avanzados (ARPA) que resolviera el problema, por lo que ARPA se contactó con centros universitarios y fabricantes de sistemas informáticos para desarrollar un sistema abierto de protocolos que dio el primer paso a lo que actualmente conocemos como TCP/IP. ARPA llevó a cabo reuniones informales de investigadores para compartir ideas y discutir los resultados de los experimentos.

La Internet global se inició alrededor de 1980 cuando ARPA comenzó a convertir las máquinas conectadas a sus redes de investigación en máquinas con el nuevo protocolo TCP/IP, la cual bautizaron como ARPANET y que rápidamente se convirtió en la columna vertebral de la nueva Internet. La transición hacia la tecnología Internet se completó en Enero de 1983, cuando la Oficina del Secretario de Defensa ordenó que todas las computadoras conectadas a redes de larga distancia utilizaran el TCP/IP. Al mismo tiempo, la Agencia de Comunicación de la Defensa (DCA), dividió ARPANET en dos redes separadas, una para la investigación futura y otra para la comunicación militar (MILNET).

22.1 Ventajas de TCP

Una cosa importante a saber sobre IP es que, por si solo, no es fiable. Suponga que diez personas de su Ethernet comienzan a transferirse la última versión de Linux del servidor de FTP del servidor de la carrera de Electricidad y Electrónica. La cantidad de tráfico generada por esto podría ser excesiva para que la maneje la pasarela, porque es demasiado lento, y posee memoria limitada. Ahora, si coincide que se envía un paquete, el servidor puede tener agotado el espacio del buffer durante un momento y por tanto no sería capaz de reenviarlo. El IP resuelve este problema simplemente descartándolo; el paquete se pierde irrevocablemente. Lo cual traslada la responsabilidad de comprobar la integridad y exactitud de los datos a los nodos extremos, y su retransmisión en caso de error.

De esto se encarga TCP, que construye un servicio fiable por encima de IP. La propiedad esencial de TCP es que usa IP para darle la impresión de una conexión simple entre dos procesos en su equipo y la máquina remota, de modo que no tiene que preocuparse de cómo y sobre que ruta viajan realmente sus datos. Una conexión TCP funciona básicamente como una tubería de doble sentido en la que ambos procesos pueden escribir y leer, puede imaginarla como una conversación telefónica. TCP identifica los extremos de tal conexión por las direcciones IP de los dos nodos implicados, y el número de los llamados puertos de cada nodo. Los puertos se pueden ver como puntos de enganche para conexiones de red. Si se hace una analogía del sistema telefónico, se puede comparar las direcciones IP con los prefijos de área (los números representarían ciudades), y los números de puerto con los códigos locales (números que representan teléfonos de personas concretas).

23 REDES LINUX

La aventura de las redes bajo linux comenzó con una serie de pequeños proyectos ya prácticamente olvidados llamados “Net-x”. El Net-1, fué un trabajo sobre redes basadas en TCP/IP.

Se creó una nueva implementación del UUCP, rescribiendo grandes partes del código, esto se conoció como Net-2. Tras una dura corrección y numerosas mejoras en el código,

cambió su nombre a Net-3 para que después saliese Linux 1.0. Esta es la versión del código base de red que se incluye actualmente en las versiones oficiales del Kernel.

23.1 Configuración del Hardware de Red

Una tarjeta Ethernet es una oblea de Silicio, atiborrada de montones de pequeños chips con números en el lomo e insertada en una ranura del PC. Para poder utilizar la tarjeta Ethernet son necesarias una serie de funciones especiales definidas en el núcleo de Linux que serán capaces de entender la forma particular de acceso al dispositivo. Esta serie de funciones son los denominados controladores del dispositivo. Linux tiene controladores de dispositivos para varias marcas de tarjetas Ethernet que son muy parecidas en su funcionamiento como ya se vio al comienzo, estas pueden ser reconocidas o instaladas en el sistema.

Un controlador se comunica con un dispositivo (tarjeta Ethernet) a través de un área de memoria de E/S que esta mapeada a los registros internos de la tarjeta y a la inversa. Todas las órdenes y datos que el núcleo envía a la tarjeta tienen que ir a través de estos registros. La memoria de E/S se describe por lo general mediante su dirección base o de comienzo. Direcciones típicas para tarjetas Ethernet son la 0x360.

El núcleo accede a un dispositivo a travéz de lo que llamamos una interface. Estas se identifican mediante nombres, que se definen internamente en el núcleo, y no son ficheros de dispositivos en el directorio /dev. Nombres típicos para los interfaces Ethernet pueden ser eth0, eth1, etc. La asignación de interfaces a los dispositivos depende normalmente del orden en el que los dispositivos son configurados; por ejemplo, la primera tarjeta Ethernet instalada será eth0, la siguiente eth1, y así sucesivamente. Una excepción a esta regla son las interfaces SLIP, que son asignadas de forma dinámica; es decir, al establecerse una conexión SLIP, se asigna una interfce al puerto serie.

Al arrancar, el núcleo muestra qué dispositivo es detectado, y qué interfaces instala. Lo siguiente es un extracto de una típica pantalla de arranque:

```
This processor honours the WP bit even when in supervisor mode. Good.
                    Floppy drive(s): fd0 is 1.44M
                    Swansea University Computer Society NET.010
                    IP Protocols: ICMP, UDP, TCP
                    PPP: version 0.2.1 (4 channels) OPTIMIZE FLAGS
TCP compresssion code copyright 1993 Regents of the University of      California
                    PPP line discipline registered.
                    SLIP: version 0.7.5 (4 channels)
CSLIP: code copyright 1993 Regents of the University of California
D10: D-Link DE-600 pocket adapter, Ethernet Address: 00:80:C8:71:76:95
Checking 386/387 coupling... Ok, fpu using excepcion 16 error reporting.
Linux version 1.1.11 (okir@monad) #3 Sat May 7 14:57:18 MT DST 2001
```

Esto indica que el núcleo ha sido compilado para TCP/IP, y se incluyen los controladores para SLIP, CSLIP, y PPP. La tercera línea antes del final indica que un adaptador de bolsillo D-Link ha sido detectado e instalado como el interface d10. Si tiene un tipo diferente de tarjeta Ethernet, el núcleo por lo general es capaz de detectar la tarjeta adecuadamente.

Lo que puede o no puede hacer una red está generalmente determinado por los protocolos que dicha red soporta y la calidad de sus implementaciones, más que por la tecnología concreta de red usada, como Ethernet, Token Ring, etc. Además, en la práctica, la elección de la tecnología de red está basada en decisiones puramente pragmáticas: qué tipo de red soporta el tipo de computadores que queremos conectar, las distancias entre los equipos, las características del cableado, etc. Por regla general, se suele usar Ethernet para sistemas de media escala, Ethernet o una red basada en el cableado de par trenzado para pequeñas redes, o redes de alta velocidad (típicamente Token Ring) para la red principal de un campus y para redes de supercomputadores, que ejecutan aplicaciones de altas prestaciones.

Por tanto, se va a asumir que ha llegado a conectar "físicamente" unas redes individuales, del tipo Ethernet o Token Ring. Ahora se enfrenta a los siguientes problemas interrelacionados: Configurar el software necesario, conectar las distintas Redes Ethernet, Token Ring, etc, para formar una única red de forma coherente y coexistente, conectar las redes al mundo exterior, o sea, Internet.

Las anteriores decisiones requieren un pequeño análisis. De hecho, la mayoría de las redes necesitan una "arquitectura", que determina la manera en que se asignan las direcciones, cómo se hace el enrutado y otras elecciones, sobre cómo los computadores interaccionan con la red. Estas decisiones deben hacerse para la red en su conjunto, preferiblemente cuando se está procediendo a su instalación inicial.

Se usará el término IP para referirse a las redes diseñadas para trabajar con TCP/IP. IP es el protocolo a nivel de red de la familia de protocolos TCP/IP, usados en Internet. Es una práctica común usar el término IP cuando se refiera a direcciones, enrutamiento y otros elementos a nivel de red. La distinción muchas veces no es lo suficientemente clara. Así que, en la práctica, los términos Internet TCP/IP e IP pueden parecer incluso intercambiables.

Los términos paquete y datagrama también suelen parecer intercambiables. Conceptualmente, un paquete es la unidad física de más bajo nivel, mientras que datagrama se refiere a la unidad de datos a nivel IP. Sin embargo, en la mayoría de las redes no se pueden distinguir porque coinciden, así es que la gente suele usar los dos términos indistintamente.

Otro término conflictivo es el de pasarela (gateway) y enrutador (router). Pasarela es el término original usado en Internet. Sin embargo, la comunidad OSI empezó a usar esta palabra con un significado distinto, así, se empezó a usar enrutador para evitar dicha ambigüedad.

24 Cortafuegos

24.1 Cortafuegos: Conceptos Básicos

Cortafuegos es el término que se emplea para referirse a una franja de bosque que se limpia de árboles, vegetación, y cualquier materia inflamable, con el fin de crear una barrera que el fuego de un posible incendio no sea capaz de atravesar. Un cortafuego es un dispositivo lógico/físico que protege una red privada del resto de la red (pública). Funcionan así:

1. Se toma un computador con capacidad de rutar (por ejemplo un PC con LiNUX)
2. Se le colocan dos interfaces (por ejemplo interfaces serie, o ethernet, o Token Ring, etc...)
3. Se deshabilita el reenvío de paquetes IP (IP forwarding)
4. Se conecta una interfaz a Internet

5. Se conecta la otra interfaz a la red que se quiere proteger

Hay dos redes distintas que comparten un computador. El computador cortafuegos, al que de ahora en adelante se llamará "cortafuegos", puede comunicarse tanto con la red protegida como con Internet. La red protegida no puede comunicarse con la Internet, y viceversa, dado que se ha deshabilitado el reenvío IP en el único computador que las conecta.

Si se quiere llegar a Internet desde la red protegida, hay que realizar primero un telnet al cortafuego, y, acceder a Internet desde él. Del mismo modo, para acceder a la red protegida desde Internet, se debe antes pasar por el cortafuego.

Este es un mecanismo de seguridad excelente contra ataques desde Internet. Si alguien quiere atacar la red protegida, primero tiene que atravesar el cortafuego. De esta manera el ataque se divide en dos pasos y, por lo tanto, se dificulta. Si alguien quiere atacar la red protegida por métodos más comunes, como el bombardeo de e-mails, o el "Gusano de Internet", simplemente no podrá alcanzarla. Con esto se consigue una protección excelente.

El mayor problema de los cortafuegos es que restringen mucho el acceso a Internet desde la red protegida. Básicamente, reducen el uso de Internet al que se podría hacer desde un terminal. Tener que entrar en el cortafuego y desde allí realizar todo el acceso a Internet es una restricción muy seria. Programas como Netscape, que requieren una conexión directa con Internet, no funcionan detrás de un cortafuego. La solución a todos estos problemas es un Servidor Proxy.

24.2 Configuración de un Cortafuegos

Lo primero es recompilar el núcleo con las opciones apropiadas.

Las opciones requeridas son:

1. Habilitar el Soporte de Red
2. Habilitar la opción de red TCP/IP (TCP/IP Networking)
3. Deshabilitar el reenvío de paquetes IP (CONFIG_IP_FORWARD)
4. Habilitar la opción de Cortafuegos IP (IP Firewalling)
5. Habilitar las cuentas IP (IP Accounting).
6. Habilitar el Soporte de Dispositivos de Red (Networking Device Support)

7. Habilitar el soporte de PPP y Ethernet (por ejemplo, aunque esto depende del tipo de interfaces que se tenga en cada caso).

Ahora, hay que recompilar, reinstalar el núcleo y rearrancar la máquina. Las interfaces deberían ser reconocidas en la secuencia de arranque.

Dado que no quiere que desde Internet se tenga acceso a las máquinas, no necesita usar direcciones reales (o válidas). Una buena elección es el rango de direcciones de clase C 192.168.2.xxx, que está designado como rango para pruebas. Nadie lo usa, y no entrará en conflicto con ninguna petición al exterior. De modo que, en esta configuración, sólo se necesita una dirección IP real. Las otras se pueden elegir libremente y de ninguna manera afectarán a la red.

Asigna la dirección IP real al puerto serie del cortafuego que usa para la conexión PPP. Asigna 192.168.2.1 a la tarjeta Ethernet del cortafuego. Asigna a las otras máquinas de la red protegida cualquier dirección del rango anterior.

Finalmente, para poder probar la configuración, lo primero es hacer ping a Internet desde el cortafuego. Ahora tendrá que intentar hacer pings entre las máquinas de la red protegida, todas deben ser capaces de hacer ping a las demás, a su vez. éstas deben ser capaces de hacer pings al cortafuego. Deberían ser capaces de hacer ping a la 192.168.2.1, no a la dirección PPP.

Luego, deberá probar hacer ping a la dirección PPP del cortafuegos desde dentro de la red protegida; no debe funcionar. Al haber asignado a la red protegida la dirección 192.168.2.0 ningún paquete será encaminado a ella por Internet, pero, en cualquier caso, es más seguro tener el reenvío de paquetes IP deshabilitado.

24.3. Seguridad para el Cortafuegos

El cortafuego no sirve si se deja vulnerable a los ataques. Primero, el fichero /etc/inetd.conf de configuración del "superservidor" (inetd) arranca un buen número de demonios servidores cuando les llega una petición, eentre ellos:

Telnet

· Talk

· FTP
· Daytime

Se debe desactivar todo lo que no se necesite. Para desactivar un servicio basta con colocar un # al comienzo de la línea que se refiera a él. Después debe teclear "kill -HUP <pid>", donde <pid> es el número de proceso de inetd. Esto hará que inetd vuelva a leer su fichero de configuración (inetd.conf) y se reinicie.

24.4 El juego de herramientas para cortafuegos de TIS

TIS ha sacado una colección de programas para facilitar la realización de cortafuegos. Básicamente, los programas hacen lo mismo que el paquete Socks, pero tiene una estrategia de diseño diferente. Mientras que Socks tiene un único programa que cubre todas las operaciones de Internet, TIS provee un programa para cada utilidad que quiera usar el cortafuego. Socks es más fácil de configurar, más fácil de compilar y permite una mayor flexibilidad. El juego de herramientas de TIS es más seguro si se quiere controlar a los usuarios de la red local. Los dos proporcionan una protección absoluta del exterior.

En linux, el filtrado de paquetes se controla a nivel del kernel. Existen módulos para el kernel que permiten definir un sistema de reglas para aceptar o rechazar los paquetes o las comunicaciones que pasan por el sistema. Estos sistemas de reglas conforman lo que se conoce como firewall o cortafuegos; en otros sistemas los firewall pueden estar implementados en software y estar desvinculados del sistema operativo pero, en el caso de linux, el firewall se puede montar a nivel de kernel y no es necesario instalar un software adicional.

24.5 IPCHAINS

En versiones de Kernel anteriores a la 2.4 se disponía de los módulos IPCHAINS para montar firewalls, así como de otros módulos de soporte para comunicaciones concretas (enmascaramiento de FTP, de IRC, Real Audio etc.). Gracias a IPCHAINS, además de poder definir normas de firewall muy precisas, también puede hacer que un linux funcione como gateway enmascarando todas las peticiones de una LAN.

24.6 IPTABLES

A partir del kernel 2.4 se está dando soporte a otro módulo para filtrado de paquetes mucho más potente que IPCHAINS, llamado IPTABLES. Para acceder a ciertos sites ftp tendrá problemas usando IPCHAINS con el kernel 2.4. A pesar de que IPCHAINS siga funcionando, ya no tendrá los antiguos módulos para solventar los problemas de acceso a servicios especiales, y se recomienda pasar a IPTABLES.

En la siguiente figura se distingue el camino que siguen los paquetes al pasar por un firewall con IPTables, y guía sobre la forma de aplicar las reglas. (En general, a lo que se enruta a través del firewall se le aplica FORWARD y a lo que va al firewall se le aplica INPUT).

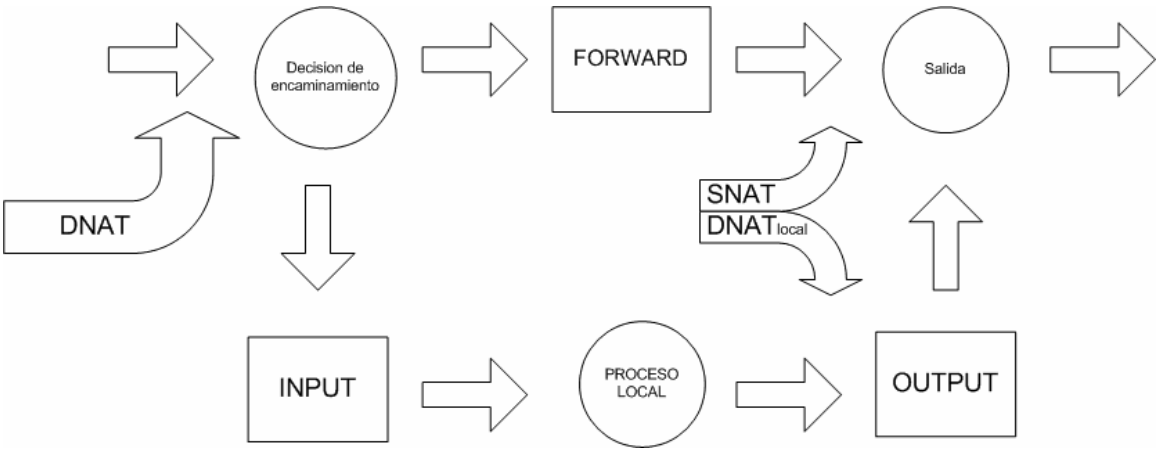


Figura 3: Diagrama de IPTables

24.6.1 Instalación de IPTABLE

Hay que cargar un módulo del kernel (se puede hacer en el propio script de firewall), y ejecutar un script de shell convencional que tiene el aspecto de un conjunto de reglas. Un script de este tipo se podría complicar y sofisticar tanto como se desea, según los requerimientos personales.

El script generalmente:
- Comienza cargando los módulos necesarios (los imprescindibles y los auxiliares, como el de ftp masquerading),
- Establece algún bit como por ejemplo el de forwarding.
- Luego borra todas las reglas actuales (flush).
- Establece las políticas por defecto para la aceptación, reenvío y salida.
- Y finalmente va aplicando todas las reglas de firewall, que varían enormemente dependiendo de las necesidades de cada red.

El orden de algunos puntos puede variar. Por lo general, una aproximación buena suele ser CERRAR todo por defecto, e ir abriendo lo que se necesite.

A continuación se mencionarán algunas diferencias respecto a IPCHAINS.

- DENY no existe, ahora sería DROP.
- MASQ y REDIRECT no existen como destinos de paquetes.
- REJECT extendidos con más opciones
- LOG con más opciones, muy útil para monitorizar y depurar

24.6.2 Elementos básicos

-Ordenes básicas:

iptables –F : efectivamente, flush de reglas
iptables –L : si, listado de reglas que se están aplicando
iptables –A : append, añadir regla
iptables –D : borrar una reglas, etc…

24.6.3 Ejemplo de regla:

#Regla que acepta conexiones al puerto 80
iptables -A INPUT -i eth0 -s 0.0.0.0/0 -p TCP --dport www -j ACCEPT

Anatomía de la regla:

iptables: comando iptables (no hay que olvidar que las reglas son un shell script)
-A: append, opción para añadir la regla
INPUT: estado del paquete (al entrar es input).
-i eth0: interfaz de red eth0
-s 0.0.0.0/0: dirección de acceso (cualquiera en este caso)
-p TCP: tipo de puerto
--dport: puerto de destino
-j ACCEPT: destino del paquete (se acepta, podría ser DROP, LOG, REJECT,..)

-Guía rápida de flags:

-s : source address.	Ej.:-s 192.168.1.0/24
-d : destino.	Ej.:-d 84.56.73.3
-p : tipo de protocolo(TCP,UDP,ICMP).	Ej.:-p TCP

--sport : puerto de origen
--dport: puerto de destino
-i = –in-interface: el interfaz por el que se entra (eth0,eth1, ppp0,…)
-o = --out-interface: el interfaz por el que se sale (eth0,eth1, ppp0,…)

-Notas:

-i se usa con reglas INPUT y FORWARD
-o se usa con reglas FORWARD y OUTPUT

A partir de estas normas básicas, conociendo la anatomía básica de una regla, y observando los ejemplos ya tiene suficiente material para tener una visión clara de IPTABLES.

24.6.4 Ejemplos de configuración.

Ejemplo de firewall simple.

```
#!/bin/sh

#########################################################
## SCRIPT de IPTABLES ##
## Este script es de ejemplo ##
## y no es el mejor ejemplo, ##
## pero funciona en RedHat 7.2 ##
## y es muy pedagógico ##
#########################################################

## Notas para usuarios de IPCHAINS:
# ipchains e iptables son módulos del kernel que
# NO pueden convivir juntos
# DENY ahora es DROP
# Los LOG se guardan de otra forma

echo -n Aplicando Reglas de Firewall...

## Instalando módulos
modprobe ip_tables
modprobe ip_nat_ftp
modprobe ip_conntrack_ftp

## Variables
IPTABLES=iptables
EXTIF="eth0" # La que va al router
INTIF="eth1" # La que va a la LAN

## Primeras reglas
$IPTABLES -P INPUT ACCEPT # INPUT se acepta por defecto MAL HECHO
```

$IPTABLES -F INPUT
$IPTABLES -P OUTPUT ACCEPT # OUTPUT se acepta por defecto
$IPTABLES -F OUTPUT
$IPTABLES -P FORWARD ACCEPT # FORWARD se acepta por defecto buf
$IPTABLES -F FORWARD
$IPTABLES -t nat -F

se deniega 80 y se guarda log (ejemplo)
$IPTABLES -A INPUT -i $INTIF -s 0.0.0.0/0 -p TCP --dport www -j LOG --log-prefix "IPTablesFW> "
$IPTABLES -A INPUT -i $INTIF -s 0.0.0.0/0 -p TCP --dport www -j DROP

Acceso al 3128 (proxy squid) desde LAN
$IPTABLES -A INPUT -i $INTIF -s 192.168.1.0/24 -p TCP --dport 3128 -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $INTIF -s 192.168.1.0/24 -p TCP --dport 3128 -j ACCEPT
El resto se descarta
$IPTABLES -A INPUT -i $INTIF -s 0.0.0.0/0 -p TCP --dport 3128 -j DROP

Acceso al 143 desde LAN
$IPTABLES -A INPUT -i $INTIF -s 192.168.1.0/24 -p TCP --dport 143 -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $INTIF -s 192.168.1.0/24 -p TCP --dport 143 -j ACCEPT

Acceso al ssh desde la LAN
$IPTABLES -A INPUT -i $EXTIF -s 213.195.64.0/24 -p TCP --dport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $EXTIF -s 213.195.64.0/24 -p TCP --dport 22 -j ACCEPT

Acceso al ssh un rango externo
$IPTABLES -A INPUT -i $EXTIF -s 213.195.64.0/24 -p TCP --dport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
$IPTABLES -A INPUT -i $EXTIF -s 213.195.64.0/24 -p TCP --dport 22 -j ACCEPT

 # el resto se descarta
$IPTABLES -A INPUT -i $EXTIF -s 0.0.0.0/0 -p TCP --dport 22 -j DROP

Acceso al puerto 25
$IPTABLES -A INPUT -i $EXTIF -s 213.191.89.0/24 -p TCP --dport 25 -j ACCEPT
$IPTABLES -A INPUT -i $INTIF -s 192.168.1.0/24 -p TCP --dport 25 -j ACCEPT
$IPTABLES -A INPUT -i $EXTIF -s 0.0.0.0/0 -p TCP --dport 25 -j DROP

FORWARD
Que me haga log de todo el forward
$IPTABLES -A FORWARD -j LOG

He aquí el forward para la LAN, una regla mágica
iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
 # Ese pedazo de bit que hay que habilitar
echo 1 > /proc/sys/net/ipv4/ip_forward

Ejemplo de firewall más completo

Este ejemplo es algo más serio, ya que la regla de input por defecto es DROP. Esta política de reglas es la más segura, ya que por defecto se denegará todo, y poco a poco se van habilitando las entradas precisas.

#!/bin/sh

SCRIPT de IPTABLES

echo -n Aplicando Reglas de Firewall...

Detenemos ipchains y quitamos el módulo
/etc/rc.d/init.d/firewall stop
rmmod ipchains

```
## Instalando módulos
modprobe ip_tables
modprobe ip_nat_ftp
modprobe ip_conntrack_ftp

## Variables
IPTABLES=iptables
EXTIF="eth1"
INTIF="eth0"

## En este caso,
## la tarjeta eth1 es la que va al ROUTER y la eth0 la de la LAN

## Primeras reglas
$IPTABLES -P INPUT DROP
$IPTABLES -F INPUT
$IPTABLES -P OUTPUT ACCEPT
$IPTABLES -F OUTPUT
$IPTABLES -P FORWARD ACCEPT
$IPTABLES -F FORWARD
$IPTABLES -t nat -F

### En principio, si las reglas INPUT por defecto hacen DROP, no será necesario
### agregar más reglas pero, si temporalmente se pasa a ACCEPT no esta de mas.

## Todo lo que viene de cierta IP se deja pasar (administradores remotos…)
$IPTABLES -A INPUT  -i $EXTIF -s 203.175.34.0/24  -d 0.0.0.0/0  -j ACCEPT

## El localhost se deja
$IPTABLES -A INPUT  -i lo -j ACCEPT
$IPTABLES -A OUTPUT -o lo -j ACCEPT

## Aceptar al exterior al 80 y al 443
```

```
# Permitir salida al 80
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 80 -m state --state ESTABLISHED -j ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport 80 -m state --state NEW,ESTABLISHED -j
ACCEPT
# Permitir salida al 443
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 443 -m state --state ESTABLISHED -j ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport 443 -m state --state NEW,ESTABLISHED -j
ACCEPT

## SALIDA SMTP - Para que el servidor se pueda conectar a otros MTA
# Permitir salida SMTP
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 25 -m state --state ESTABLISHED -j ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport 25 -m state --state NEW,ESTABLISHED -j
ACCEPT

## SALIDA FTP - Para que el servidor se pueda conectar a FTPs
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 21 -m state --state ESTABLISHED -j ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport 21 -m state --state NEW,ESTABLISHED -j
ACCEPT
# ftp activo
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 20 -m state --state ESTABLISHED,RELATED -j
ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --dport 20 -m state --state ESTABLISHED -j ACCEPT
# ftp pasivo
$IPTABLES -A INPUT  -i $EXTIF -p tcp --sport 1024:65535 --dport 1024:65535  -m state --state
ESTABLISHED -j ACCEPT
$IPTABLES -A OUTPUT -o $EXTIF -p tcp --sport 1024:65535 --dport 1024:65535  -m state --
state ESTABLISHED,RELATED -j ACCEPT

## El acceso al 19720 desde fuera, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0  -d 0.0.0.0/0 -p tcp --dport 19720 -j DROP
```

```
## El acceso al 19720 desde dentro, ACEPTADO
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.0/24 -p tcp --dport 19720 -j ACCEPT

## El acceso al 19721 desde fuera, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0  -d 0.0.0.0/0 -p tcp --dport 19721 -j DROP

## El acceso al SSH desde fuera, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0  -d 0.0.0.0/0 -p tcp --dport 1972 -j DROP

## El acceso al SMTP desde dentro, permitido.
$IPTABLES -A INPUT  -i $INTIF -p tcp --dport 25 -j ACCEPT
## El acceso al SMTP desde fuera, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0  -d 0.0.0.0/0 -p tcp --dport 25 -j DROP

## Acceso al 80 desde el interior ACCEPTADO PARA DOS IPs
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.11/32 -p tcp --dport 80 -j ACCEPT
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.54/32 -p tcp --dport 80 -j ACCEPT
## Acceso al 80 desde el interior DENEGADO PARA EL RESTO
#$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.0/24  -p tcp --dport 80 -j DROP

## Acceso al PROXY
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.0/24 -p tcp --dport 8082 -j ACCEPT
$IPTABLES -A INPUT  -i $INTIF -s 192.168.10.0/24 -p tcp --dport 8082 -j ACCEPT
$IPTABLES -A INPUT   -s 127.0.0.0/8 -p tcp --dport 8082 -j ACCEPT

# Desde el exterior denegado
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0 -p tcp --dport 8082 -j DROP

## Acceso a POP3 e IMAP desde el EXTERIOR, DENEGADO
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.0/24 -p tcp --dport 110 -j ACCEPT
$IPTABLES -A INPUT  -i $INTIF -s 192.168.9.0/24 -p tcp --dport 143 -j ACCEPT
$IPTABLES -A INPUT  -i $INTIF -s 192.168.10.0/24 -p tcp --dport 110 -j ACCEPT
$IPTABLES -A INPUT  -i $INTIF -s 192.168.10.0/24 -p tcp --dport 143 -j ACCEPT
```

```
## Acceso a POP3 e IMAP desde el EXTERIOR, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0 -p tcp --dport 110 -j DROP
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0 -p tcp --dport 143 -j DROP
## Acceso al 8082 desde fuera, DENEGADO
$IPTABLES -A INPUT  -i $EXTIF -s 0.0.0.0/0 -p tcp --dport 8082 -j DROP

## FORWARD
# Que me haga log de todo el forward
#$IPTABLES -A FORWARD -j LOG

# He aquí el forward

## Norma general
##iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

iptables -t nat -A POSTROUTING -o eth1 -s 192.168.9.11/32 -j MASQUERADE
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.9.16 -j MASQUERADE
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.9.54/32 -j MASQUERADE
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.9.0/24 -j MASQUERADE
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.10.0/24 -j MASQUERADE

# Habilitar el forwarding
echo 1 > /proc/sys/net/ipv4/ip_forward
```

24.7 Reglas de protección

En esta breve sección se listarán algunas reglas para proteger el equipo y, por extensión la red, de ciertos ataques muy habituales como el smurf y otras formas de inundación y DoS. ¿Es recomendable usar todas estas normas? Según como administre el nivel de paranoia. Nota: hay que dar valores concretos a las variables $

```
# Deshabilitar broadcast
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_broadcasts
# Deshabilitar  el ping… quizá discutible.
/bin/echo "1" > /proc/sys/net/ipv4/icmp_echo_ignore_all
# Deshabilitar la redirección del  ping
/bin/echo "0" > /proc/sys/net/ipv4/conf/all/accept_redirects

# Registrar los accesos extraños, paquetes falseados, etc..
/bin/echo "1" > /proc/sys/net/ipv4/conf/all/log_martians
# Anti-flooding o inundación de tramas SYN.
iptables -N syn-flood
iptables -A INPUT -i $IFACE -p tcp --syn -j syn-flood
iptables -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN
iptables -A syn-flood -j DROP

# Guardar los accesos con paquetes fragmentados, recurso utilizado para descartar paquetes
# servidores y otras falencias (bug en Apache por ejemplo)
iptables -A INPUT -i $IFACE -f -j LOG --log-prefix "Fragmento! "
iptables -A INPUT -i $IFACE -f -j DROP

# Anti-spoofing (falseo de ip origen)
iptables -A INPUT  -i $IFACE -s $IPADDR -j DROP
iptables -A INPUT  -i $IFACE -s $CLASS_A -j DROP
iptables -A INPUT  -i $IFACE -s $CLASS_B -j DROP
iptables -A INPUT  -i $IFACE -s $CLASS_C -j DROP
iptables -A INPUT -i $IFACE -s $CLASS_D_MULTICAST -j DROP
iptables -A INPUT -i $IFACE -s $CLASS_E_RESERVED_NET -j DROP
iptables -A INPUT  -i $IFACE -d $LOOPBACK -j DROP
iptables -A INPUT -i $IFACE -d $BROADCAST -j DROP
(bien explicado en el script de linuxguruz)

# Proxy Transparent: peticiones al puerto 80 redirigir al SQUID(3128)
iptables -t nat -A PREROUTING -p tcp -s 0.0.0.0/0 --dport 80 -j REDIRECT --to 3128
```

25 Servidores Proxy

25.1 Servidores Proxy: Conceptos básicos

Los servidores proxy son servicios que permiten el acceso directo a Internet desde detrás de un cortafuego. Funcionan abriendo un socket en el servidor y permitiendo la comunicación con Internet a través de él, por ejemplo, si un computador estuviera dentro de la red protegida y quisiera ver el Web con Netscape, pondría un servidor proxy en el cortafuego. El servidor proxy estaría configurado para hacer que las peticiones de conexión de su computador al puerto 80 de otra máquina, se conectará a su puerto 1080, y él mismo establecería una conexión con el puerto 80 de la máquina deseada. A partir de entonces reenviaría todos los datos de esa conexión a la otra máquina. TIA o TERM son dos programas con los que se puede redirigir un puerto. Lo importante de los servidores proxy es que, bien configurados, son completamente seguros. No dejan que nadie ingrese a través de ellos.

25.2 Instalación del Servidor Proxy

El servidor proxy requiere software adicional. Éste se puede conseguir en:

Ftp://sunsite.unc.edu/pub/LiNUX/system/Network/misc/socks-Linux-src.tgz

Algo importante que hay que advertir es que hay que añadir el servidor proxy al /etc/inetd.conf. Se debe añadir la línea:

socks stream tcp nowait nobody /usr/local/etc/sockd sockd, para decir a inetd que arranque el servidor cuando se le pida.

El programa socks necesita dos ficheros de configuración distintos. Uno en el que se le dice qué accesos están permitidos, y otro para dirigir las peticiones al servidor proxy apropiado. El fichero de control de acceso debe residir en el servidor. El fichero de rutado debe residir en todas las máquinas Unix.

25.3 El Fichero de Control de Acceso

El fichero de acceso se llama "sockd.conf". Debería contener dos tipos de líneas: las de permiso, que contienen "permit" y las de prohibición, que contienen "deny". Cada línea tendrá tres palabras:

· El identificador (permit/deny)
· La dirección IP
· El modificador de dirección

El identificador es o permit (permitir) o deny (denegar). El modificador de dirección es también un número de cuatro octetos. Funciona como una máscara de red. Hay que verlo como 32 bits (unos o ceros). Si el bit es uno, el bit correspondiente de la dirección que se comprueba debe coincidir con el bit correspondiente del campo de dirección IP.
Por ejemplo, si la línea es:

permit 192.168.2.23 255.255.255.255

entonces, admitirá sólo direcciones IP en las que coincidan todos los bits de 193.168.2.23, esto es, sólo ella misma.
La línea:

permit 192.168.2.0 255.255.255.0

admitirá todas las direcciones desde la 192.168.2.0 hasta la 192.168.2.255, la subred de clase C completa. No se debería tener la línea:

permit 192.168.2.0 0.0.0.0

dado que permitiría cualquier dirección.

Así es que, primero, permitirá todas las direcciones que quiera permitir, y después prohibirá el resto. Para permitir a cualquiera del rango 192.168.2.xxx, las líneas:

permit 192.168.2.0 255.255.255.0
deny 0.0.0.0 0.0.0.0

25.4 El Fichero de rutado

El fichero de rutado de socks tiene el nombre de "socks.conf". El fichero de rutado tiene la función de decir a los clientes de socks cuándo usar socks y cuándo no. Por ejemplo, en la red, la máquina 192.168.2.3 no necesita usar socks para comunicarse con la 192.168.2.1 (el cortafuego), ya que tiene una conexión directa vía Ethernet (por ejemplo). Hay tres tipos de entradas:

· deny
· direct
· sockd

Deny (denegar) le dice a socks que peticiones debe rechazar. Esta entrada tiene los mismos tres campos que en sockd.conf, identificador, dirección, y modificador. La entrada direct dice para qué direcciones no se debe usar socks. Éstas son todas las direcciones a las que se puede llegar sin usar el servidor proxy. Nuevamente hay tres campos: identificador, dirección, y modificador.

La entrada sockd dice cuál es la máquina en la que corre el servidor de socks. La sintaxis es:

sockd @=<lista de servidores> <dirección IP> <modificador>

La entrada "@=" permite poner las direcciones IP de una lista de servidores proxy. En el ejemplo sólo se usa un servidor proxy, pero se puede tener muchos para admitir una carga mayor y conseguir redundancia frente a fallos.

25.5 Trabajar con un Servidor Proxy

Para que las aplicaciones funcionen con el servidor proxy, hay que "sockificarlas". Será necesario tener dos telnets distintos, uno para la comunicación directa, y uno para la

comunicación a través del servidor proxy. Socks viene con instrucciones de cómo "sockificar" un programa. Por esta razón deberá cambiar el nombre a todos los programas de la red protegida y sustituirlos por los "sockificados". Así "finger" pasará a ser "finger.orig", "telnet" a "telnet.orig", etc... . Se debe dar a conocer a socks todo esto en el fichero include/socks.h . Algunos programas gestionan el rutado y el "sockificado" ellos mismos.

25.6 Inconvenientes de los Servidores Proxy

Un servidor proxy es ante todo un dispositivo de seguridad. Se usa para aumentar el número de máquinas con acceso a Internet cuando se tienen pocas direcciones IP. Un servidor proxy permite un mayor acceso desde dentro de la red protegida al exterior, pero mantiene el interior completamente inaccesible desde el exterior. Esto significa que no habrá conexiones archie, ni talk, ni envío directo de correo a los computadores dentro de la red.

FTP crea otro problema con los servidores proxy. Cuando se hace un ls, o un get, el servidor de FTP establece una conexión con la máquina cliente y envía la información por ella. Un servidor proxy no lo permitirá, así que el FTP no funciona bien. Además, un servidor proxy es lento, debido a la gran sobrecarga.

26 Routing Avanzado

El núcleo Linux, a partir de la versión 2.4 ofrece una interfaz que permite implementar herramientas profesionales de alto nivel en cuanto a la gestión del tráfico de paquetes IP. Permite hacer túneles IP, tablas de routing múltiples, reserva de ancho de banda, multicasting, proxy ARP y mucho más. Esas funcionalidades estaban hasta ahora sólo disponibles en routers propietarios de gama alta y de precio casi prohibitivo. El núcleo Linux permite implementarlas de modo más seguro, más económico y con mayor rendimiento, además de desarrollar sus propias herramientas específicas.

A partir de la versión 2.4 del núcleo Linux, está disponible un socket llamado NETLINK que permite implementar en espacio usuario (user space) código de gestión de paquetes y tráfico IP. Teóricamente, este socket, por su naturaleza, permite implementar cualquier código.

Sin embargo, se centrará en el código escrito por Alexey Kuznetsov y disponible bajo el nombre 'iproute'. Está accesible en ftp://ftp.inr.ac.ru/ip-routing

A lo largo de esta unidad se verán algunas de las cosas que permite hacer este código:

• unificar los comandos relacionados con la gestión del tráfico IP, sea de redes, de interfaces, etc.
• monitoreo de los periféricos, direcciones y rutas.
• gestión de tablas ARP
• uso de tablas de routing múltiples
• creación de túneles IP
• reserva de ancho de banda

26.1 Configuración del sistema

Para que funcione el iproute, necesita configurar el núcleo para que provea el socket NETLINK que interesa. En /usr/src/linux/.config vienen bastantes opciones que le van a permitir adaptar el kernel a sus necesidades:

```
CONFIG_NETLINK=y
CONFIG_RTNETLINK=y
# CONFIG_NETLINK_DEV is not set
CONFIG_NETFILTER=y
CONFIG_NETFILTER_DEBUG=y
# CONFIG_FILTER is not set
CONFIG_UNIX=y
CONFIG_INET=y
# CONFIG_IP_MULTICAST is not set
CONFIG_IP_ADVANCED_ROUTER=y
CONFIG_RTNETLINK=y
CONFIG_NETLINK=y
CONFIG_IP_MULTIPLE_TABLES=y
CONFIG_IP_ROUTE_FWMARK=y
```

```
CONFIG_IP_ROUTE_NAT=y
CONFIG_IP_ROUTE_MULTIPATH=y
CONFIG_IP_ROUTE_TOS=y
CONFIG_IP_ROUTE_VERBOSE=y
CONFIG_IP_ROUTE_LARGE_TABLES=y
# CONFIG_IP_PNP is not set
CONFIG_NET_IPIP=m
CONFIG_NET_IPGRE=m
# CONFIG_ARPD is not set
# CONFIG_INET_ECN is not set
```

26.2 El comando ip

Actualmente se usarán varios comandos para gestionar el tráfico IP y todo lo que le rodea: interfaces, rutas, túneles,... Se decidió unificar el conjunto y proveer así un solo comando con una sintaxis coherente y global. El comando se llama 'ip' y tiene la siguiente sintaxis:

```
ip [OPTIONS] OBJECT [COMMAND [ARGUMENTS]]
```

OPTIONS

Son varias opciones que influyen el comportamiento general de la herramienta. Todas las opciones empiezan por el carácter '-' y se pueden abreviar. Algunas opciones son:

- -s, -stats, -statistics obtener más información
- -f, -family especifica que familia de protocolo usar: inet, inet6 o link.
- -r, -resolve imprime nombres DNS en lugar de direcciones de host

OBJECT

Es el objeto que queremos manejar o del cual buscamos informaciones. He aquí algunos ejemplos que también pueden ser abreviados:

- link,l -- periférico de red

• address,a -- dirección (IPv4 o IPv6) de periférico
• route,r -- entrada en la tabla de routing
• rule,ru -- regla en la base de datos de política (policy database)
• maddress,maddr -- dirección multicast
• tunnel,t -- tunnel sobre IP

COMMAND

Es el comando que se aplica al objeto. Se puede abreviar también:

• add,a -- añadir un objeto
• del,d -- borrar un objeto
• set,s -- ajustar un objeto
• show,list,l -- ver un objeto

Mensajes de error
Algunos posibles errores son:
• Wrong syntax of command line -- problema de sintaxis
• el núcleo devuelve un error a una petición NETLINK -- En este caso, ip imprime el mensaje de error prefijado por "RTNETLINK answers:"
• Cannot open netlink socket: Invalid value -- Netlink no está configurado en el núcleo
• Cannot talk to rtnetlink: Connection refused -- RTNETLINK no está configurado en el núcleo
• Cannot send dumb request: Connection refused -- RTNETLINK no está configurado en el núcleo
• RTNETLINK error: Invalid argument -- CONFIG_IP_MULTIPLES_TABLES no está configurado en el núcleo

ip link -- manejar las interfaces
ip link set -- cambiar los atributos de la interfaz

• dev NAME: especifica de que interfaz se trata
• up/down: cambiar el estado de la interfaz
• name NAME: cambiar nombre de la interfaz

• mtu NUMBER: cambiar MTU de la interfaz
ejemplo:

ip link set dummy up
ip link show -- ver los atributos

• dev NAME: mostrar la interfaz especificada
• up: mostrar solo las interfaces ’up’

ejemplos:

[telsur@telsur:~]$ ip l l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
[telsur@telsur:~]$

La primera línea de cada entrada da un número único a la interfaz, su nombre (que puede ser modificado), así como la información sobre el estado de la interfaz. La segunda línea da antecedentes sobre el tipo de interfaz de que se trata, la dirección de la interfaz a nivel de la capa ’layer’ (en el caso de ethernet, la dirección MAC).
La opción -s le permite ver estadísticas de la interfaz:

[telsur@telsur:~]$ ip -s l l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
RX: bytes packets errors dropped overrun mcast
1368991 5872 0 0 0 0
TX: bytes packets errors dropped carrier collsns
1368991 5872 0 0 0 0
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc

```
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
RX: bytes packets errors dropped overrun mcast
1789685066 1975826 99656 0 0 0
TX: bytes packets errors dropped carrier collsns
1619835989 2304565 69 16 4 1762589
[telsur@telsur:~]$
```

Los parámetros son similares al antiguo comando 'ifconfig'.

ip address -- gestión de las direcciones de interfaz

ip addr permite ver las direcciones de interfaz, añadir nuevas direcciones o borrarlas. Es importante destacar que a partir de iproute, las interfaces físicas y las direcciones son totalmente disociadas, eso significa que una interfaz puede tener varias direcciones sin necesitad de crear un alias como ocurría en el caso anterior.

ip addr show: ver direcciones de protocolo

```
telsur:~# ip a l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
inet 192.168.2.71/24 brd 192.168.2.255 scope global eth0
telsur:~#
```

ip addr add: añadir nueva dirección

```
telsur:~# ip a l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
inet 192.168.2.71/24 brd 192.168.2.255 scope global eth0
telsur:~# ip a a 10.0.0.1 dev eth0
telsur:~# ip a l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
inet 192.168.2.71/24 brd 192.168.2.255 scope global eth0
inet 10.0.0.1/32 scope global eth0
telsur:~#

ip addr del: borrar una dirección

telsur:~# ip a d 10.0.0.1 dev eth0
telsur:~# ip a l
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc
pfifo_fast qlen 100
link/ether 00:05:1c:01:b1:33 brd ff:ff:ff:ff:ff:ff
inet 192.168.2.71/24 brd 192.168.2.255 scope global eth0
telsur:~#

26.3 Gestión de tablas ARP

Las tablas ARP establecen enlaces entre las capas de protocolo y de 'link' (en caso de una red local Ethernet, sería entre dirección IP y dirección MAC). Cada host de una subred necesita conocer la dirección física de los demás para poder enviar paquetes a los destinatarios adecuados. Estas direcciones se almacenan en un caché ARP. En caso de no conocer la dirección MAC, se hace una petición de broadcast (por ejemplo cuando se enciende el equipo) que sería cómo "Hola! Quién es la dirección IP w.x.y.z?". La máquina con tal dirección IP contesta: "Yo soy la máquina w.x.y.z y mi MAC es 00: ad: f3:b1:22:4e". Entonces, la máquina almacena esta MAC en su caché.

El Objeto 'neighbour' del comando ip gestiona el cache ARP. Estos son algunos ejemplos:

```
telsur:~# ip neigh ls
192.168.2.5 dev eth0 lladdr 00:c0: ca: 15:80:9c nud reachable
192.168.2.34 dev eth0 lladdr 00:05:1c:01:9d:c3 nud delay
192.168.2.72 dev eth0 lladdr 00:05:1c:01:6c:a9 nud reachable
192.168.2.1 dev eth0 lladdr 00:c0: ca: 15:81:07 nud reachable
192.168.2.3 dev eth0 lladdr 00:01:02: ad: 08: da nud stale
192.168.2.70 dev eth0 lladdr 00:40:f6:2c:27:13 nud reachable
192.168.2.21 dev eth0 lladdr 00:50: fc: 42:07:b4 nud reachable
192.168.2.52 dev eth0 lladdr 00:05:1c:01:5e:1b nud delay
telsur:~#
```

Para borrar una entrada:

```
telsur:~# ip n d 192.168.2.52 dev eth0
telsur:~# ip n l
192.168.2.5 dev eth0 lladdr 00:c0:ca:15:80:9c nud reachable
192.168.2.34 dev eth0 lladdr 00:05:1c:01:9d:c3 nud reachable
192.168.2.72 dev eth0 lladdr 00:05:1c:01:6c:a9 nud reachable
192.168.2.1 dev eth0 nud failed
192.168.2.3 dev eth0 lladdr 00:01:02:ad:08:da nud reachable
192.168.2.70 dev eth0 lladdr 00:40:f6:2c:27:13 nud reachable
```

192.168.2.21 dev eth0 lladdr 00:50:fc:42:07:b4 nud reachable
192.168.2.52 dev eth0 lladdr 00:05:1c:01:5e:1b nud reachable

Que ocurrió?, realmente la entrada correspondiente no desaparecerá de inmediato. Se quedará hasta que el último cliente que la usa la “libere”

Los otros comandos para añadir o cambiar una entrada son:

• ip neigh add
• ip neigh change

26.4 Tablas de routing múltiples

Ip le permite trabajar con tablas de routing múltiples, una gran novedad en la gestión de tráfico IP. Hasta ahora había tenido, por cada sistema, una tabla de routing única, que define a qué interfaz está conectada una red, donde se encuentra el gateway, etc ... Gracias a iproute, puede trabajar con varias tablas de routing a la vez y elegir que tabla usar según las características del paquete IP.

Imagine por ejemplo, que tiene un router con dos interfaces de conexión a Internet. Una interfaz RDSI más lenta pero barata y una interfaz ADSL rápida, pero más cara. La ventaja es que se puede decidir, gracias a iproute, que interfaz será usada, según los paquetes que hay que enviar. Por ejemplo, le puede decidir que los paquetes SMTP saldrán por la interfaz RDSI lenta (no hay prisa). Al revés, si tiene aplicaciones de videoconferencia, deseará que vayan por la interfaz ADSL rápida. En este caso, la política de routing se basa en el puerto de destino de los paquetes IP.

También puede basar la política de routing sobre la dirección IP de origen de los paquetes. Sería el caso si quiere dar prioridad a determinados servicios de una empresa. Por ejemplo, la dirección de la empresa usará la conexión ADSL rápida, mientras el servicio de mecanografía verá su tráfico dirigido por la interfaz lenta.

En realidad, puede basarse en muchos parámetros para establecer su política de routing: IP de origen, IP de destino, puerto de origen, puerto de destino, protocolo usado, TOS e interfaz de llegada.

Hay que destacar que para realizar comprobaciones sobre protocolos IP y puertos de transporte, hay que usar el sistema conjuntamente con ipchains que nos provee fwmark, un sistema para marcar paquetes. Por defecto, hay 3 tablas de routing en la base de datos 'routing policy database':

```
[telsur@telsur:~]$ ip ru l
0: from all lookup local
32766: from all lookup main
32767: from all lookup default
[telsur@telsur:~]$
```

El antiguo comando 'route' le enseña la tabla 'main'. Las dos otras son nuevas.
La tabla 'local' es especial, no puede ser borrada y tiene la prioridad más alta (0). Se usa para direcciones locales y de broadcast. La tabla 'main' es la tabla clásica que le devuelve el antiguo comando 'route'. Se puede borrar o cambiar. La tabla 'default' esta vacía y se reserva para procesos de post-routing (si las reglas anteriores no coinciden). También se puede borrar.

```
[telsur@telsur:~]$ ip r l table main
195.96.98.253 dev ppp2 proto kernel scope link
src 212.64.78.148
212.64.94.1 dev ppp0 proto kernel scope link
src 212.64.94.251
192.168.2.0/24 dev eth0 proto kernel scope link
src 192.168.2.71
127.0.0.0/8 dev lo scope link
default via 212.64.94.1 dev ppp0
[telsur@telsur:~]$
```

Ejemplo sencillo de routing según origen
Imagine que las dos interfaces de red que se han mencionado antes son:
212.64.94.1 para el tráfico de alta velocidad 195.96.98.253 para el tráfico lento.
A un Usuario A lo va a redireccionar a través de la conexión lenta. Generará una regla que se llama 'Usuario_A':

```
# echo 200 Usuario_A >> /etc/iproute2/rt_tables
# ip rule add from 192.168.2.35 table Usuario_A
# ip rule ls
0: from all lookup local
32765: from 192.168.2.35 lookup Usuario_A
32766: from all lookup main
32767: from all lookup default
```

Esta regla especifica que todo el tráfico que viene con IP de origen 192.168.2.35 usa la tabla de routing llamada Usuario_A. Necesita entonces crear la misma tabla Usuario_A:

```
# ip route add default via 195.96.98.253 dev ppp2 table Usuario_A
# ip route flush cache
```

A continuación verá los identificadores que se pueden usar en la base de datos:

- from: determina la origen del paquete
- to: determina el destino del paquete
- iis: determina la interfaz de llegada
- tos: determina el valor de TOS
- fwmark: determina el valor de 'marca' del paquete (puesto por iptables por Ej., ver el ejemplo siguiente)

Es importante no confundir 'tablas de routing' y 'reglas'. Las reglas apuntan a tablas de routing, varias reglas pueden apuntar a la misma tabla de routing. De la misma manera, una tabla de routing puede no tener ninguna regla apuntando a ella sin dejar de existir.

En el ejemplo previo, ha dirigido los paquetes según su dirección IP de origen. Si quiere, por otra parte, basar su política sobre el tipo de tráfico (correo, web, vídeo), usará la herramienta de marcación de paquetes proveída por netfilter (iptables).

Siguiendo con los datos previos, verá ahora como desviar su tráfico de web saliente hacia la interfaz rápida (212.64.94.1, ver arriba).

Marcará los paquetes que tienen como destino el puerto 25 (SMTP) con un número '1':

```
# iptables -A PREROUTING -i eth0 -t mangle -p tcp --dport 25 \
-j MARK --set-mark 1
```

Ahora, dirigirá los paquetes marcados ’1’ a una tabla de routing especifica, en este caso, la llamamos web.out:

```
# echo 201 mail.out >> /etc/iproute2/rt_tables
# ip rule add fwmark 1 table web.out
# ip rule ls
0: from all lookup local
32764: from all fwmark 1 lookup web.out
32765: from 192.168.2.35 lookup Pedro
32766: from all lookup main
32767: from all lookup default
```

Sólo le hace falta ahora crear la tabla de routing:

```
#ip route add default via 195.96.98.253 dev ppp0 table web.out
```

La regla de marca que usará es muy simple, sólo se basa en el puerto de destino del paquete. Obviamente, con la flexibilidad que le ofrece iptables, pode hacerlo mucho más complejo, añadiendo excepciones, etc.

26.5 ip tunnel -- Túneles

Sin necesidades de usar herramientas externas y complicadas como PPTP, iptunnel, etc. Puede crear túneles cifrados o no, con el mismo comando ip. Existen tres tipos de túneles disponibles: IPIP (IP sobre IP), sit y GRE (el protocolo desarrollado por Cisco).

Se trata realmente de encapsular paquetes IP en otros paquetes IPv4 y mandarlos a través de una infraestructura IP.

ip tun [add|change|delete] -- ’abrir/cambiar/cerrar’ un túnel

Los argumentos posibles son:

• name NOMBRE -- selecciona el nombre del túnel
• mode MODO -- hay 3 modos disponibles: ipip, sit y gre

El modo ipip corresponde a un simple túnel IP sobre IP. Se encapsulan los paquetes sin más. El modo sit se usa para túneles IPv6. El modo gre corresponde a los túneles GRE especificados por la compañía Cisco, que son túneles IP sobre IP cifrados.

• remote DIRECCION -- dirección de 'salida' del tunel
• local DIRECCION -- dirección local de 'entrada' del tunel
• dev PERIFERICO-- nombre del periférico a través del que se envían los paquetes

Por ejemplo, para crear un tunel IPv6 sobre IPv4,

```
ip tunl add MiTunel mode sit remote 192.31.7.104\
local 192.203.80.142
```

ip tun show -- ver los túneles

```
telsur:~# ip tu ls
tunl0: ip/ip remote any local any ttl inherit nopmtudisc
mitun: ip/ip remote 192.168.2.5 local 192.168.2.71 ttl inherit
telsur:~#
```

o con estadísticas

```
telsur:~# ip -s tu ls mitun
mitun: ip/ip remote 192.168.2.5 local 192.168.2.71 ttl inherit
RX: Packets Bytes Errors CsumErrs OutOfSeq Mcasts
0 0 0 0 0 0
TX: Packets Bytes Errors DeadLoop NoRoute NoBufs
0 0 0 0 0 0
telsur:~#
```

27. Utilidades para la Administración de Red

27.1 Tcpdump

Tcpdump es un programa cuya utilidad principal es analizar el tráfico que circula por la red. Se apoya en la librería de captura *pcap*, la cual presenta una interfaz uniforme y que esconde las peculiaridades de cada sistema operativo a la hora de capturar tramas de red.
Lo primero que se debe averiguar cuando se usa tcpdump, son las interfaces que quiere escuchar. Por defecto cuando se ejecuta sin parámetros, bajo Linux se pone a escuchar en la interfaz eth0.

Para averiguar las interfaces en cualquier Unix se recurre al comando *ifconfig -a* el cual da una lista de las interfaces que tiene, así como sus parámetros de configuración.

```
telsur:~# ifconfig -a
eth0      Link encap:Ethernet  HWaddr addr
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:646760 errors:0 dropped:0 overruns:0 frame:0
           TX packets:449673 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:100
           Interrupt:5 Base address:0x2c20

eth1      Link encap:Ethernet  HWaddr addr
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:1321583 errors:0 dropped:0 overruns:0 frame:0
           TX packets:1778135 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:100
          Interrupt:9 Base address:0x3000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:3924  Metric:1
          RX packets:39747 errors:0 dropped:0 overruns:0 frame:0
```

```
          TX packets:39747 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
telsur:~#
```

En el ejemplo anterior están borradas las direcciones ip.

Usando estos datos, si quiere escuchar en la interfaz eth0, usará *tcpdump -i eth0*. Cuando esté leyendo la red, puede que no le interese que tcpdump intente resolver los nombres de las máquinas (pueden que no estén dadas de alta en el DNS), para ello dispone de la opción *-n*.

Para establecer la longitud de los datos que captura tcpdump usará *-s len*, donde *len* es la longitud que le interesa. Por defecto tcpdump sólo captura los primeros 68 bytes, lo cual es útil si lo único que se desea son las cabeceras IP, TCP o UDP, ahora en caso de estar buscando protocolos como el NFS los datos se truncan. En ese caso, puede ajustar la longitud de la captura a la MTU del medio que está usando con esta opción. Por ejemplo para capturar toda la trama ethernet podemos *usar -s 1500*.

Si quiere manejar una gran cantidad de información, puede usar *-v,-vv,-vvv*, aumentando el grado de información con cada una de las opciones.

Si quiere imprimir el contenido del paquete, puede usar la opción *-x*. Si además quiere que se imprima en ASCII el contenido de los paquetes puede usar *-X*. La longitud que imprime viene determinada por la opción *-s* o los 68 bytes que usa la captura por defecto.

Puede trabajar offline con tcpdump. Si quiere grabar su captura para posteriormente leerla y analizarla use la opción *-w file* donde *file* es el nombre del fichero donde quiere grabar la captura de datos. Posteriormente puede leer y analizar offline con *-r file*.

27.1.1 Interpretando la salida

Es necesario aclarar que la salida depende del protocolo que esté analizando. Todas las capturas de tcpdump tienen como primer campo una marca de tiempo, que indica cuando ha sido capturado el paquete.
Las peticiones arp aparecen de la siguiente manera:

```
18:33:49.908612 arp who-has 192.168.1.2 tell 192.168.1.1
18:33:49.908691 arp reply 192.168.1.2 is-at 0:2:a5:ee:ec:10
```

En este caso, la máquina 192.168.1.1 pregunta por la dirección ethernet 192.168.1.2 (se supone que ambas máquinas son de la misma subred). Como ve, la 192.168.1.2 responde, en este caso, vea los valores numéricos puesto que está usado la opción *-n*. Las mayúsculas indican la máquina por la cual se está preguntando, y las minúsculas la máquina que hace la pregunta.

```
18:33:49.908612 arp who-has MAQUINA tell otra
18:33:49.908691 arp reply MAQUINA is-at 0:2:a5:ee:ec:10
```

La línea general de un paquete TCP es como sigue:

```
src > dst: flags [dataseq ack window urgent options]
```

En principio src, dst y flags están siempre presentes. Los otros dependiendo del tipo de conexión TCP que se trate. El significado de dichos parámetros es:

src: Dirección y puerto origen. En caso de no especificar el parámetro *-n* se intenta resolver el nombre vía DNS y el se busca el nombre del puerto vía (normalmente en los Unix en /etc/services).

dst: Dirección y puerto destino, exactamente igual que el caso anterior.

flags: Indica los flags de la cabecera TCP. Puede ser un ., cuyo significado es que no hay flags, o bien una combinación de S (SYN), F (FIN), P (PUSH), W (reducción de la ventana de congestión), E (ECN eco).

dataseq: El número de secuencia del primer byte de datos en éste segmento TCP. El formato es primero: ultimo(n), que significa que desde a primero a último (sin incluir último) hay un total de n bytes de datos. Atento cuando hay segmentos con SYN, que también ocupa un número del espacio de secuencia.

ack: El número de aceptaciones. Indica el número siguiente de secuencia que se espera recibir. Es necesario resaltar que los SYN también se asienten.

win: Tamaño de la ventana de recepción.

urgent: Existen datos urgentes.

options: Indica la existencia de opciones. En caso de que existan, van entre < y >.

En el siguiente ejemplo, puede ver:

```
rtsg.1023 > csam.login: S 768512:768512(0) win 4096 <mss 1024>
csam.login > rtsg.1023: S 947648:947648(0) ack 768513 win 4096 <mss 1024>
rtsg.1023 > csam.login: . ack 1 win 4096
rtsg.1023 > csam.login: P 1:2(1) ack 1 win 4096
csam.login > rtsg.1023: . ack 2 win 4096
rtsg.1023 > csam.login: P 2:21(19) ack 1 win 4096
csam.login > rtsg.1023: P 1:2(1) ack 21 win 4077
csam.login > rtsg.1023: P 2:3(1) ack 21 win 4077 urg 1
csam.login > rtsg.1023: P 3:4(1) ack 21 win 4077 urg 1
```

Lo anterior, simula una conexión originada por la máquina rtsg con destino a csam, con el servicio rlogin, y el significado de las líneas anteriores es:

1. Inicio de conexión de rtsg -> csam SYN ISN 768512 ventana de 4096
2. SYN de csam -> rtsg ISN 947648 ventana de 4096 ACK del SYN anterior.
3. ACK del SYN enviado por csam. No hay flags
4. 1 byte de datos de rtsg -> csam. Flag PUSH activado., (los números de secuencia son relativos al ISN a menos que se especifique la opción -S, en cuyo caso los números de secuencia se imprimen de manera absoluta).
5. ACK del byte de datos anterior por parte de csam.
6. 19 bytes de datos de rtsg a csam.
7. csam manda 1 byte de datos a rtsg, y manda el ACK de los 19 bytes enviados por rtsg. La ventana de recepción ha bajado en 19 bytes. Flag PUSH
8. csam envía un byte de datos urgente. Flag PUSH.
9. Idem anterior.

Un paquete UDP se imprime de la siguiente manera:

origen.srcport > destino.dsrpot: udp len

origen: Nombre o dirección origen.
srcport: Puerto origen.
destino: Nombre o dirección destino.
dstport: Puerto destino
len: Longitud de los datos de usuario.

Ejemplo:

12:35:21.457350 10.10.109.10.1025 > 192.168.1.2.1345: udp 121 [ttl 1]

En algunos casos, puede interpretar protocolos que vayan encapsulados en los paquetes UDP, como NFS o DNS. El grado de detalle en la interpretación de estos protocolos dependerá del grado de detalle (controlado con la opción -v) que quiera darle.

Los datagramas fragmentados se indican con una expresión al lado de los mismos entre paréntesis, por ejemplo:

(frag id:size@offset+) (frag id:size@offset)

id: es el identificador de fragmento.
Size: tamaño del fragmento.
Offset: posición del fragmento en el datagrama original. Si existe signo + al final de offset significa que aún quedan más fragmentos. En caso de ausencia, que es el último, los datos del protocolo del nivel superior, sólo se imprimen en el primer fragmento.

27.1.2 Filtros

Es lo más importante que permite hacer tcpdump. Un filtro es una expresión que va detrás de las opciones y que le permite seleccionar los paquetes que esta buscando. En ausencia de ésta, tcpdump volcará todo el tráfico que vea el adaptador de red seleccionado.

La expresión que se usa para definir el filtro, tiene una serie de primitivas y tres posibles modificadores a las mismas. Esta expresión será verdadera o falsa y hará que se imprima o no el paquete de datos.

Los 3 modificadores posibles son:

tipo. Puede ser host, net o port. Indican respectivamente una máquina, por ejemplo el host 192.168.1.1, una red completa, net 192.168, o un puerto concreto, *port* 22. Por defecto se asume el tipo host.
dir. Especifica desde o hacia donde se va a mirar el flujo de datos. Tiene *src* o *dst* y puede combinarlos con *or* y *and*. Para el caso de de protocolos punto a punto puede sustituir por inbound o outbound. Por ejemplo, si quiere la dirección de destino 10.10.10.2 y la de origen 192.168.1.2, el filtro sería *dst* 10.10.10.2 *and src* 192.168.1.2. Si se quiere que sea la dirección destino 192.168.1.1 o la dirección origen 192.168.1.2, sería *dst* 192.168.1.1 *or src* 192.168.1.2. Pueden seguirse combinando con la ayuda de paréntesis o las palabras *or* y *and*. Si no existe se supone *src or dst*. Por supuesto, esto se puede combinar con los modificadores anteriores.
proto. En este caso es el protocolo que se quiere capturar, puede ser tcp, udp, ip, ether (en este caso captura tramas a nivel de enlace, arp (peticiones arp), rarp (petciones reverse-arp), fddi (para redes FDDI, pero realmente el encapsulado es igual al ether). Hay otros niveles de enlace

para redes Decnet y lat, por ejemplo. Siempre puede combinar expresiones con ayuda de paréntesis.

A continuación se dan las primitivas que pueden usarse. Lo que aparece entre [y] es opcional, y el signo | significa "o". El resto se tiene que poner si queremos poner el filtro con el comportamiento.

[*dst*|*src*] host máquina. Es decir, si la dirección destino u origen del paquete es máquina lo cual puede ser una dirección IPv4 (o IPv6 si se ha compilado soporte para el mismo), o un nombre del DNS. Si quiere restringir a dirección destino puede hacerlo con *dst*. Para dirección origen *src*.

Ejemplos:
Capturar el trafico cuya IP origen sea 192.168.1.1
tcpdump src host 192.168.1.1

Capturar todo el tráfico cuya dirección origen o destino sea 192.168.1.2
tcpdump host 192.168.1.2

ether src|dst|host edir. Este filtro funciona si la dirección origen (*src*), la destino (*dst*) o cualquiera de las dos (*host*) coincide con *edir*. Hacer notar que *src*, *dst* o *host* debe ser especificado en forma obligatoria.

Ejemplos:
Capturar el tráfico con destino a la dirección ethernet 0:2:a5: ee: ec: 10.
tcpdump ether dst 0:2:a5:ee:ec:10

Capturar el tráfico que vaya a la máquina cuya dirección MAC es 0:2:a5: ee: ec: 10.
tcpdump ether host 0:2:a5:ee:ec:10

gateway máquina. Se Aplica para el caso en que el paquete use entremedio una máquina como router. *máquina* debe estar definida en /etc/ethers y /etc/hosts. Los paquetes que cumplen con esa condición son aquellos que tienen como dirección ethernet destino máquina, pero ni la dirección IP destino u origen es *máquina*.

[*dst*|*src*] *net red*. Se aplica en caso de que la red de la dirección destino, origen o ambas sea *red*. El parámetro *red* puede ser una dirección numérica (por ejemplo 192.168.1.0) o bien un nombre que se resuelve a dirección, en los Unix, con ayuda del /etc/networks. Es necesario destacar que también se admite el clásico direccionamiento CIDR. Puede especificar una máscara poniendo *red* como *net red mad mascara* o bien usar /, *net red/bits*. Hacer notar que el uso de *net... mask* no es compatible con direcciones IPv6. Si quiere hacer referencia a la red destino use *dst* como prefijo. Para la red origen use *src*.

Ejemplos:
Capturar todo el tráfico cuya red destino sea 192.168.1.0.
tcpdump dst net 192.168.1.0

Capturar todo el tráfico cuya red origen sea 192.168.1.0/28
tcpdump src net 192.168.1.0 mask 255.255.255.240
tcpdump src net 192.168.1.0/28

Capturar todo el tráfico con origen o destino en la 10.0.0.0/24
tcpdump net 10.0.0.0/24
tcpdump net 10.0.0.0 mask 255.255.255.0

[*dst*|*src*] *port puerto*. Aplica en caso de que el puerto (ya sea udp o tcp) coincida con *puerto*. Si no se especifica *dst* o *src*, se tomará tanto el puerto origen como destino. Si quiere restringir a destino use *dst,* y a origen use *src*. El puerto es un valor numérico entre 0-65535 o bien un nombre que en Unix se resuelve a través del /etc/services.

Ejemplos:
Capturar todo el tráfico con destino al puerto 23
tcpdump dst port 23

Capturar todo el tráfico con destino u origen puerto 80
tcpdump port 23

less longitud. Aplica en caso de que el tamaño del paquete sea menor o igual a *longitud*.

greater longitud. Aplica en caso de que el tamaño del paquete sea mayor o igual que *longitud*.

ip proto protocolo. En este caso escucha el protocolo que se le indique. El protocolo puede ser *icmp*, *icmp6*, *igmp* (internet group managent protocol), *igrp* (interior gateway routing protocol), *pim* (protocol independent multicast), *ah* (IP Authentication header), *esp* (encapsulating security payload), *udp* o *tcp*. En caso de usar *icmp*, *udp* o *tcp* hay que separar el protocolo, poniendo un \, es decir, *ip proto* *icmp*.
Por comodidad se disponen los alias tcp, udp e icmp que equivalen a *ip proto tcp or ip6 proto tcp*, etc.

Ejemplos:
Capturar el todo los paquetes icmp
tcpdump ip proto \\ip
(en Unix hay que separar el \).

Capturar todo el tráfico udp
tcpdump ip proto \\udp
tcpdump udp

ip6 proto protocolo. Aplica si es un paquete de IPv6 con el protocolo *protocolo*.

ip6 protochain protocolo. Es un número que en Unix puede leerse en /etc/protocols. En este caso lo que se busca es que dentro de las diferentes cabeceras que puede tener un paquete IPv6 una de ellas sea el protocolo especificado.

ip protochain protocolo. Igual que el caso anterior, pero para IPv4.

ether broadcast. Aplica si la trama capturada va dirigida hacia la dirección de difusión ethernet. La palabra *ether* es opcional.

ip broadcast. Aplica si el paquete va dirigido a la dirección de difusión de IP. Esta dirección se comprueba si es todo 0 o 1, o bien se comprueba la dirección local de la subred.

ether multicast. Aplica si la trama va dirigida a una dirección multicast ethernet.

ip multicast. Aplica si el paquete va dirigido a una dirección multicast IP.

ip6 multicast. Aplica si el paquete va dirigido a una dirección multicast IPv6.

ether proto protocolo. Aplica si el protocolo que contiene la trama es de tipo *protocolo* Los protocolos son *ip*, *ip6*, *arp*, *rarp*, *atalk*, *aarp*, *decnet*, *sca*, *lat*, *mopdl moprc* e *iso*. Además, estos nombres son identificadores que deben de ser separados con \.

Sin embargo, hay una serie de alias que hacen más cómoda la expresión en los filtros. Dichas expresiones son *ip*, *ip6*, *arp*, *rarp*, *aarp*, *decnet* e *iso*, siendo equivalentes a *ether proto ip*, *ether proto ip6*, etc.

Ejemplos:
Capturar todo tráfico arp
tcpdump -n ether proto \\arp
tcpdump -n arp
(el alias es más cómodo)

Capturar todo tráfico ip
tcpdump -n ether proto \\ip
tcpdump -n ipi

vlan [*vlanid*]. Aplica si la trama capturada es un paquete 802.1Q VLAN. Hacer notar que esto modifica el resto de la interpretación del paquete capturado, en especial, los desplazamientos a partir de los cuales comienza a decodificar los protocolos, ya que se asume, que está capturando paquetes que viajan en tramas VLAN. Por último, si está presente el parámetro *vlanid*, sólo se mostrarán aquellos paquetes que vayan a la VLAN *vlanid*.

Combinando los filtros

Se pueden combinar las expresiones anteriores con los ayuda de los operadores *not*, *and* y *or* (corresponden a la negación, el y lógico y el o lógico, dando lugar a filtros más complejos. Puede usar también los equivalentes del lenguaje C: !, && o ||.

Ejemplos:
Capturar todo el tráfico Web (TCP port 80)
tcpdump tcp and port 80

Capturar el todas las peticiones DNS
tcpdump udp and dst port 53

Capturar el tráfico al puerto telnet o ssh
tcpdump tcp and \(port 22 or port 23\)
(los "\" son para escapar en el shell de Unix)

Capturar todo el tráfico excepto el web
tcpdump tcp and not port 80

27.1.3 Filtros Avanzados

Tcpdump permite hacer filtros a mano, indicando que bytes de la trama quiere encontrar y como los quiere interpretar. Cuando quiere definir filtros de esta manera la expresión general es:

expr relop expr

Donde, *relop* puede ser cualquiera de las operaciones de relación de C: >,<, >= <=, = y !=. *expr* es una expresión aritmética compuesta por una serie de números enteros, los operadores binarios de C, (+, -, *, /, & y |), un operador de longitud, *len*, y una serie de palabras reservadas que le permiten el acceso a los diferentes paquetes de datos (*ether*, *fddi*, *tr*, *ip*, *arp*, *rarp*, *tcp*, *udp*, *icmp* e *ip6*). Para acceder a los datos dentro de un paquete, debe usar los modificadores anteriores y una expresión entera. Opcionalmente puede especificar el tamaño de los datos que accede.
proto [*expr* : *tam*]

Así por ejemplo, el primer byte de la trama ethernet será ether[0], la primera palabra será ether[0:2]. El parámetro tam puede ser 1 (por defecto y no es necesario especificarlo), 2 o 4.

A menos en la página de manual de la versión 3.6.2 se hace notar que cuando se especifica tcp, udp u otro protocolo de nivel superior se hace referencia a IPv4. Esta limitación sin embargo no aparece en OpenBSD.

En caso de usar tcp[índice] o udp[índice], implícitamente se aplica una regla para averiguar si es un paquete fragmentado, es decir, usando la notación de estos filtros ip[0:2] & 0x1fff = 0. udp[0] o tcp[0] se refieren al primer byte de la cabecera UDP o TCP.

27.2 MRTG

El Graficador de Tráfico Multi Enrutador (Multi Router Traffic Grapher, MRTG) es una herramienta para monitorear la carga de tráfico en los enlaces de una red. MRTG genera páginas HTML las cuales contienen gráficos GIF lo que provee un representación visual en tiempo real de este tráfico.

Principales Características

Portable

El MRTG trabaja sobre la mayoría de las plataformas UNIX y sobre windows NT.

Perl

El MRTG está escrito en Perl y viene con la fuente completa.

Portable SNMP

El MRTG usa una implementación de SNMP altamente portable escrita completamente en Perl gracias a Simon Leinen. No es necesario instalar ningún paquete de SNMP externo.

Soporte para SNMPv2c

El MRTG puede leer los nuevos contadores de 64bit de SNMPv2.

Identificación de Interfaces Confiable

Las interfaces de los enrutadores pueden ser identificadas por su dirección IP, Descripción y dirección Ethernet además del número de interfaz normal.

Bitácoras (logs) de tamaño constante

Las bitácoras del MRTG no aumentan. Gracias al uso de un algoritmo único de consolidación de datos.

Configuración Automática

MRTG viene con un conjunto de herramientas de configuración las cuales hacen la configuración muy simple.

Gráficos libres de GIF

Los gráficos son generados directamente en formato PNG, usando la biblioteca GD.

Adaptabilidad

La apariencia de las páginas web producidas por el MRTG son altamente configurables.

RRDtool

MRTG tiene relaciones intrínsecas para usar RRDtool.

27.2.1 Detalles de MRTG

MRTG consiste en un programa en Perl que usa SNMP para leer los contadores de tráfico de sus enrutadores y de un rápido programa en C el cual archiva los datos de tráfico y crea imágenes que representan el tráfico en la conexión de red monitoreada. Esos gráficos se insertan en páginas web que pueden ser vistas desde cualquier browser.

Además de una vista diaria detallada, el MRTG crea también representaciones visuales para el tráfico de los últimos siete días, las cuatro últimas semanas y los últimos doce meses. Esto es posible pues, MRTG mantiene un archivo de todos los datos que ha obtenido del enrutador. Este archivo es consolidado automáticamente, así es que no crece con el tiempo, pero contiene todos los datos relevantes del tráfico de los últimos dos años. Todo esto se realiza de una manera eficiente. Por lo tanto, podemos monitorear 200 o más enlaces de red desde cualquier máquina Unix.

MRTG no está limitado al monitoreo de tráfico, es posible monitorear cualquier variable SNMP que usted elija. Puede hasta usar un programa externo para recolectar datos que serán monitoreados por MRTG. En particular se usa MRTG en la Telefónica del Sur para monitorear eventos como Carga del Sistema, Logueo de Sesiones, disponibilidad de modems y más.

MRTG permite además acumular dos o más fuentes de datos en un único gráfico.

27.2.2 Instalación de MRTG

Descargue de Internet las fuentes de MRTG

```
mkdir /home/install/
cd /home/install/
```

wget "http://people.ee.ethz.ch/~oetiker/webtools/mrtg/pub/mrtg-2.9.17.tar.gz"

Descomprima el tarball
cd /usr/src
tar -zxvf /home/install/mrtg-2.9.17.tar.gz
ln -s mrtg-2.9.17 mrtg

Compile MRTG
cd /usr/src/mrtg
./configure
make
make install

Prepare directorios y ficheros necesarios
mkdir /home/httpd/www.midominio.com/html/mrtg
touch /etc/mrtg.cfg

El fichero de configuración de MRTG, que es /etc/mrtg.cfg, puede generarlo con la ayuda de un programa que trae MRTG, /usr/local/mrtg-2/bin/cfgmaker de la siguiente forma:

```
/usr/local/mrtg-2/bin/cfgmaker \
 --global 'WorkDir: /home/httpd/www.midominio.com/html/mrtg' \
 --global 'Options[_]: bits,growright' \
 --output /etc/mrtg.cfg \
 secreto@34.25.12.154
```

Pruebe que funciona MRTG:
/usr/local/mrtg-2/bin/mrtg /etc/mrtg.cfg

Programe el cron para que cada 5 minutos se refresquen las estadísticas:
00,05,10,15,20,25,30,35,40,45,50,55 * * * * /usr/local/mrtg-2/bin/mrtg /etc/mrtg.cfg

Visualice los resultados:

http://www.midominio.com/mrtg/34.25.12.154_2.html

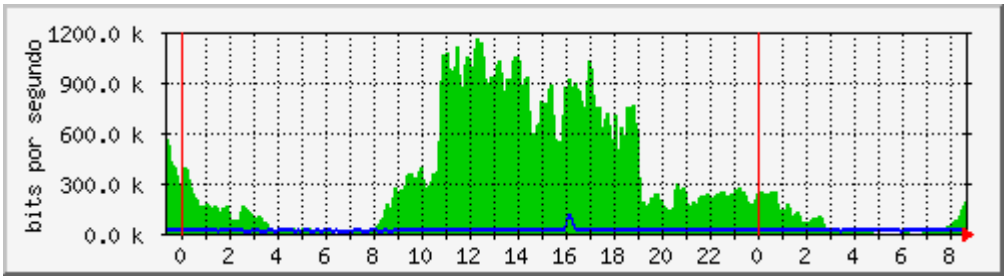


Figura 4: Ejemplo MRTG

CONCLUSIONES

Durante el presente trabajo se puso de manifiesto las principales características que hace de Linux un Sistema Operativo portable.

Se demostró lo fácil e intuitivo que es instalar Linux en un PC reacondicionado con Hardware básico. Si bien un dominio cabal del sistema abarca varias horas de entrenamiento, esta guía es un punto de partida para comprender el funcionamiento de Linux.

La mayoría de los servicios de red están soportados para Linux en forma segura a diferencia de Windows NT.

La adaptación que tiene éste sistema operativo sirvió para controlar el tráfico de información con MRTG desde Telefónica del Sur a sus clientes en forma eficiente.

Clientes corporativos de Telefónica del Sur, al igual que cualquier empresa relacionada a temas de interconectividad, pueden disponer de estos servicios ya que son gratuitos. Por lo cual, los dispositivos bajo Linux son una solución a bajo costo.

El proceso de configuración de Firewalls bajo Linux es muy sencillo e intuitivo, demás está resaltar las comparaciones precio/rendimiento entre equipos comerciales y una máquina reacondicionada en Linux.

El proceso de formación de técnicos en ambientes de conectividad es más práctico y fácil de comprender utilizando Linux. De esta manera el entrenamiento, se realiza de una manera más completa.

Gracias al apoyo de los laboratorios, se logró tener una referencia de los pasos a seguir para levantar un servidor Linux con todos sus procesos controlados.

REFERENCIAS BIBLIOGRÁFICAS

Mancill, Tony. 2001. Linux Routers: A Prime for Network Administrators. Prentice Hall, Inc. New York USA.

Takett, Jack., S. Burnett.2000. Linux. 4ta Ed.; Prntice Hall. Madrid. España.

Zieger, Robert. 2000. Guia Avanzada Firewalls Linux. Prentice Hall Iberia. Madrid. España.

Alexey, S.,N. Kuznetsov. IP Command Reference:
http://snafu.freedom.org/linux2.2/docs/ip-cref/ip-cref.html ,

Bert Hubert et al., Linux 2.4 Advanced Routing HOWTO:
http://www.linuxdoc.org/HOWTO/Adv-Routing-HOWTO.html

Comparativa de configuración ipfwadm, ipchains, iptables:
http://www.tldp.org/LDP/nag2/x-087-2-firewall.example.html

Firewalls
http://www.linuxguruz.org/iptables/scripts/rc.firewall_010.txt

Guías COMO de: http://es.tldp.org/htmls/comos.html

TCPDUMP, www.tcpdump.org

MRTG, www.mrtg.org

Sendmail, www.sendmail.com

RFC 821, RFC 974, RFC 1123, http://www.rfc-editor.org/

Proyecto Netfilter Seguridad a través de filtrado de paquetes
http://www.netfilter.org

ANEXO : Guías de Laboratorio

Guías de Laboratorio:

Laboratorio N°1:

Objetivos:

- Preparar la instalación de Linux; reuniendo toda la información necesaria que permita disponer de manera óptima de los recursos tanto de Hardware como de Software.
- Decidir que distribución de Linux usar y el tipo de instalación, dependiendo del uso que se le dará al equipo (Servidor o Workstation).
- Definir claramente las particiones en el disco duro para administrarlo de forma efectiva.
- Comprender como se interactúa con los comandos básicos en los entornos de usuarios

Pasos:

1. Reunir toda la información del Equipo, por ejemplo: Memoria, disco duro, procesador, video, etc.
2. Seleccionar la distribución de acuerdo al criterio del administrador.
3. Definir el tipo de instalación a utilizar, por ejemplo: servidor, workstation, Servidor/Workstation, etc.
4. Utilizar fdisk de Linux para crear las particiones. (pag 36)
5. Activar memoria Swap. (pag 41)
6. Comenzar a instalar por paquetes en forma manual. (pag 44)
7. Entorno de Usuario:
    a. Crear cuentas de Usuarios. (pag 46)
    b. Cambiar de Consola Virtual. (pag 47)
    c. Analizar árbol de Directorios. (pag 51)
    d. Usar comandos básicos. (pag 62)
8. Elaborar un informe en el cual detallará el trabajo realizado.

Laboratorio N°2:

Objetivos:

Una vez ya instalado Linux; y estar familiarizado con su entorno, deberá ser capaz de:

- Utilizar y analizar las entradas y salidas estándar y sus redirecciones.
- Comprender la utilización de tuberías para extender el uso de comandos.
- Cambiar los permisos de ficheros y los enlaces hacia los ficheros.
- Analizar los procesos y tareas que están ejecutando en el Sistema.

Pasos:

1. Crear al menos dos cuentas de usuario y cambiar las palabras de contraseña. (pag 46)
2. Tomar el archivo README.TXT y manipularlo con los comandos ed, more, cp, mv y rm. (pag 60)
3. Crear Ficheros con mkdir. (pag 64)
4. Definir la jerarquía de los sistemas de Ficheros y cambiar los permisos de éstos de acuerdo al ususario. (pag 81)
5. Usar el carácter comodín para: copiar, mover y eliminar un grupo de ficheros. (pag 72)
6. Usar Cat y Sort para entradas y salidas estándar. Definir en que lugar se pueden utilizar estos elementos en forma práctica.
7. Realizar ejemplos de tuberías para extender los comandos ya conocidos. (pag 75)
8. Ver cuales son los procesos que se están ejecutando. (pag 88)
9. Elaborar un informe en el cual detallará el trabajo realizado.

Laboratorio N°3:

Objetivos:

Deberá ser capaz de:

- Comenzar a levantar los servicios o demonios.
- Entender conceptos de Redes para levantar los servidores que se montan sobre TCP/IP
- Levantar servidores Web, DNS, Correo y FTP.
- Comprender las instrucciones de los ficheros de configuración para cada servicio.
- Conocer los comandos para la gestión de los Servidores.

Pasos:

1. Instalar una tarjeta de red, definir la dirección IP del equipo de acuerdo a la red local donde usted esté. (pag 134)
2. Instalar los paquetes necesarios para levantar un servidor Web con Apache. (pag 96)
3. Instalar los paquetes que se requieran para levantar un servidor de Nombres con Bind8.(pag 102)
4. Instalar los paquetes necesarios para levantar un servidor de correo con Sendmail. (pag 116)
5. Hacer las modificaciones necesarias en los ficheros httpd.conf, srm.conf y access.conf para la correcta configuración, puesta en marcha y seguridad para poner en funcionamiento un servidor Web básico. (pag 96)
6. Levantar un servidor de correo sendmail básico.(pag 122)
7. Levantar un servidor FTP básico. (pag 124)
8. Para cada uno de los servidores, bajar y levantar los servicios e indicar como verificar las configuraciones.
9. Elaborar un informe en el cual detallará el trabajo realizado.

Laboratorio N°4:

Objetivos:

Deberá ser capaz de:

- Comprender de forma acabada el enrutamiento bajo Linux.
- Levantar servicios avanzados sobre TCP/IP, Tablas de arp, definición de rutas y Túneles.
- Configurar su computador para realice funciones de cortafuego, aplicando políticas de filtrado.
- Configurar un servidor Proxy con control de acceso y reenvío de paquetes.
- Utilizar programas de monitoreo para la administración de red (TCPDUMP y MRTG).

Pasos:

1. Colocar en funcionamiento un cortafuegos básico, para lo cual deberá:
    a) Habilitar el Soporte de Red y la opción de red TCP/IP (TCP/IP Networking) (pag 155)
    b) Deshabilitar el reenvío de paquetes. (pag 137)
    c) Habilitar la opción de Cortafuegos IP (IP Firewalling). (pag 137)
    d) Poner en funcionamiento las cuentas IP (IP Accounting). (pag 137)
    e) Montar el Soporte de Dispositivos de Red (Networking Device Support) y el soporte de PPP y Ethernet. (pag 137)

2. Habilitar IPTABLES y configurar un cortafuego que deniegue todo el tráfico por defecto y habilitar las entradas por puertos para WEB, FTP y correo. (pag 140)
3. Montar un servidor Proxy editando los permisos en el archivo sockd.conf (pag 151)
4. Mediante el comando IP especifique 2 redes a través de interfaces distintas que se vean entre ellas. (pag 156)
5. Con el programa TCPDUMP, capture todo el tráfico proveniente de una de las redes con destino a un puerto en particular, por ejemplo FTP o WEB. (pag 167)
6. Levantar un servidor MRTG que monitoreará el tráfico SNMP de un Router de frontera CISCO. (pag 178)
7. Elaborar un informe en el cual detallará el trabajo realizado.