



Universidad Técnica Federico Santa María

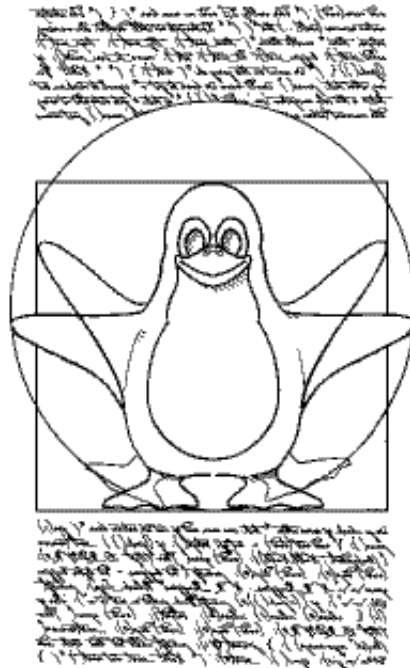


# Curso Linux Avanzado

## Departamento Informática U.T.F.S.M.

Mauricio Vergara Ereche  
mave@inf.utfsm.cl

Carlos Molina Ramírez  
penny@inf.utfsm.cl



VALPARAÍSO, DICIEMBRE 2004

# Índice

<b>1. Introducción</b>	<b>5</b>
1.1. Agradecimientos . . . . .	5
<b>2. Instalación del Sistema</b>	<b>6</b>
2.1. Métodos de instalación . . . . .	6
2.2. Tareas previas . . . . .	6
2.2.1. Su Hardware . . . . .	6
2.2.2. Espacio en disco . . . . .	7
2.2.3. Cuanto espacio es requerido? . . . . .	7
2.3. Arrancar el Instalador . . . . .	8
2.4. Definición del esquema de particiones . . . . .	8
2.4.1. Particionamiento Automatico . . . . .	9
2.4.2. Particionamiento Manual con Disk Druid . . . . .	9
2.5. Preconfigurando el Sistema . . . . .	10
2.5.1. Gestor de Arranque . . . . .	10
2.5.2. Configuración de red . . . . .	12
2.5.3. Configuración del cortafuegos . . . . .	12
2.5.4. Selección del soporte del idioma . . . . .	12
2.5.5. Configuración del huso horario . . . . .	12
2.5.6. Configuración de la contraseña de <i>root</i> . . . . .	13
2.5.7. Configuración de la autenticación . . . . .	13
2.6. Selección de paquetes . . . . .	13
2.6.1. Selección individual de paquetes . . . . .	13
2.6.2. Dependencias no satisfechas . . . . .	14
2.7. Finalizando la instalación . . . . .	14
2.7.1. Tarjeta de Vídeo . . . . .	14
2.7.2. Monitores . . . . .	14
2.7.3. Configuración de X-Window . . . . .	14
2.8. Tareas posteriores a la instalación . . . . .	15
<b>3. Configuración de dispositivos</b>	<b>15</b>
3.1. Teclado . . . . .	15
3.2. Mouse . . . . .	16
3.3. Vídeo . . . . .	17

3.3.1. Instalando X.org . . . . .	17
3.3.2. Configurando X.org . . . . .	17
3.3.3. nVidia . . . . .	19
3.4. Sonido . . . . .	20
3.5. Adaptadores de red . . . . .	20
3.6. Impresoras . . . . .	22
3.7. Tuning del sistema . . . . .	22
<b>4. Trabajando como root</b>	<b>25</b>
4.1. Precauciones . . . . .	25
4.2. Trabajando con servicios . . . . .	25
4.2.1. Niveles de ejecución . . . . .	26
4.2.2. Utilidades de los niveles de ejecución . . . . .	27
4.3. Servicios básicos . . . . .	27
4.3.1. Intervención del sistema . . . . .	27
4.3.2. Estableciendo los servicios necesarios . . . . .	28
4.3.3. Apagando servicios . . . . .	28
<b>5. El kernel en Linux</b>	<b>31</b>
5.1. Comandos de manejo de módulos . . . . .	31
5.2. El sistema de archivos inicial . . . . .	32
<b>6. Procesos y Señales</b>	<b>33</b>
6.1. Procesos . . . . .	33
6.1.1. Ejecutando procesos en segundo plano . . . . .	33
6.1.2. Listando los procesos del sistema . . . . .	34
6.2. Señales . . . . .	34
6.3. Creando y verificando el sistema de archivos ext3 . . . . .	35
6.3.1. Migrando los sistemas de archivos de ext2 a ext3 . . . . .	35
6.3.2. Reparar un sistema de archivos ext3 que está dañado . . . . .	36
6.4. Sistemas de Volúmenes Lógicos (LVM) . . . . .	36
6.4.1. Ocupando LVM . . . . .	37
<b>7. Automatizando tareas de administración con BASH</b>	<b>39</b>
7.1. Principios de programación con BASH . . . . .	39
7.1.1. Variables de entorno . . . . .	39

---

7.1.2. Uso de las comillas . . . . .	40
7.1.3. Tests . . . . .	41
7.1.4. Estructuras de control . . . . .	41
7.1.5. Globbing . . . . .	42
7.2. Creación de Scripts . . . . .	43
7.2.1. Hola mundo! . . . . .	43
7.2.2. Un ejemplo más complejo . . . . .	44
<b>8. El sistema de paquetes Red Hat Package Manager (RPM)</b>	<b>45</b>
8.1. Instalando y Desinstalando Paquetes . . . . .	45
8.1.1. Consultas . . . . .	45
8.1.2. Verificación . . . . .	46
8.1.3. Instalar y Actualizar . . . . .	46
8.1.4. Desinstalar . . . . .	47
8.2. Herramientas de adquisición de alto nivel . . . . .	47
8.2.1. APT - Advance Package Tool . . . . .	47
8.2.2. YUM - YellowDog Updater Modified . . . . .	49
8.2.3. Construyendo Repositorios . . . . .	50
8.3. Construyendo un RPM simple . . . . .	52
8.3.1. Construyendo a Partir de un SRC-RPM . . . . .	52
8.3.2. Construyendo a Partir de un tar.gz . . . . .	52

# 1. Introducción

## 1.1. Agradecimientos

Este documento, es fruto del esfuerzo de varias personas que han colaborado en su desarrollo a lo largo de los últimos 3 años, durante los cuales se ha desarrollado estos cursos en el Departamento de Informática de la Universidad Técnica Federico Santa María. La mayoría de ellos estudiantes, apoyados por el incentivo constante del profesor Horst von Brand y el espacio de trabajo que les ha dado el Laboratorio de Computación *LabComp*, han permitido que el movimiento Linux haya crecido de muy buena manera, apoyando a mucha gente y llegando incluso tener una destacada presencia mundial, al ser sede de 2 de los Encuentros Nacionales de Linux, realizados en Octubre del 2003 y Octubre del 2004<sup>1</sup>.

Se hacen especiales agradecimientos a todos aquellos que participaron de alguna u otra forma en la confección de este documento:

- Carlos Massoglia Lillo.
- Carlos Molina Ramírez,
- Horst von Brand,
- José Miguel Herrera,
- Luis Arévalo Reyes,
- Marcelo Olgún Mena,
- Mauricio Araya López,
- Mauricio Vergara Ereche,
- Nicolás Troncoso Carrère,
- Roberto Bonvallet Carrasco,
- Verónica Ramírez Duarte,

El Objetivo de este documento<sup>2</sup> es profundizar los conocimientos de los usuarios básicos de Linux. Este documento trata principalmente sobre la distribución Fedora core 3 Linux, en la cual se harán las sesiones prácticas de este curso. Se asume que el lector posee nociones básicas del sistema operativo Linux, por lo cual ciertos detalles básicos son pasados por alto en pro de una profundización de los temas vistos en los cursos básicos de Linux.

---

<sup>1</sup><http://www.encuentrolinux.cl>

<sup>2</sup>Se le agradecerá al atento lector el reportar cualquier error u omisión en la confección de este documento a sus autores en; [mave007@inf.utfsm.cl](mailto:mave007@inf.utfsm.cl) o [penny@inf.utfsm.cl](mailto:penny@inf.utfsm.cl)

## 2. Instalación del Sistema

El instalador de Fedora core 3, más conocido como *Anaconda*, es el programa que nos permitirá instalar el sistema operativo en nuestro disco duro, a la vez que configura los dispositivos que posea nuestro PC. Este instalador tiene dos interfaces de instalación; una interfaz gráfica y una interfaz modo texto. La instalación no depende del tipo de interfaz que se escoja, por lo que se pueden usar indistintamente. Cabe decir que para computadoras más pequeñas (poca ram, y procesadores lentos) es recomendable usar el instalador en modo texto.

El instalador se cambiará automáticamente de modo gráfico a modo texto si la computadora no es capaz de ejecutar adecuadamente el instalador en modo gráfico.

### 2.1. Métodos de instalación

Fedora core Linux provee los siguientes métodos de instalación.

- **CD-ROM**

Si posee un lector de CD-ROM y tiene el CD-ROM de Fedora core, puede utilizar este método. Necesitará una imagen booteable (comúnmente conseguida en un CD-ROM o un pendrive) para arrancar. También puede usar discos de arranque PCMCIA.

- **Disco duro**

Si ha copiado las imágenes ISO de Fedora core en el disco duro local, puede utilizar este método. Necesitará un disquete de arranque. También se pueden utilizar disquetes de controlador PCMCIA.

- **Imagen NFS**

Si está realizando la instalación desde un servidor NFS utilizando imágenes ISO o una imagen réplica de Fedora core, puede utilizar este método. Necesitará una imagen booteable de arranque por red. También se pueden utilizar disquetes de controlador PCMCIA.

- **FTP**

Si está realizando la instalación directamente desde un servidor FTP, utilice éste método. Necesitará una imagen booteable de arranque por red. También se pueden utilizar disquetes de controlador PCMCIA.

- **HTTP**

Si está realizando la instalación directamente desde un servidor Web HTTP, utilice este método. Necesitará una imagen booteable de arranque por red. También se pueden utilizar disquetes de controlador PCMCIA.

### 2.2. Tareas previas

En esta sección detallaremos algunas tareas que es conveciente realizar antes de la intalación de un sistema Linux.

#### 2.2.1. Su Hardware

Si tiene otro sistema operativo es importante que anote el modelo de cada pieza de hardware, si bien los Kernel modernos tienen soporte para la mayoría del hardware existente, existen algunos con

especificaciones propietarias<sup>3</sup> los cuales deberan ser configurados una vez terminada la instalación.

Si no tiene un sistema operativo instalado, recurra a los manuales para tener nota del hardware que compone a su PC.

### 2.2.2. Espacio en disco

Casi todos los sistemas operativos (SO; OS, operating system en inglés) modernos utilizan particiones de discos, y Fedora core no es una excepción. Cuando instale Fedora core, tendrá que trabajar con particiones de disco.

Si Fedora core va a compartir su sistema con otro SO, necesitará estar seguro de tener espacio disponible suficiente en su(s) disco(s) duro(s) para la instalación.

El espacio de disco destinado a Fedora core debe estar separado del espacio utilizado por otros sistemas operativos que puedan estar instalados en su sistema, como por ejemplo Windows, OS/2, o incluso una versión diferente de Linux. Al menos dos particiones (/ y swap) deben estar dedicadas a Fedora core.

Antes de comenzar el proceso de instalación, deberán reunirse al menos una de las condiciones siguientes:

- Su ordenador deberá tener espacio sin particionar para la instalación de Red Hat Linux.
- Deberá contar con una o más particiones que pueda borrar para conseguir más espacio libre para instalar Fedora core.

### 2.2.3. Cuanto espacio es requerido?

Fedora core ofrece distintos esquemas de instalacion, para acomodar mejor las necesidades de cada usuario. Los esquemas posibles son siguientes<sup>4</sup>:

- **Escritorio personal**

Una instalación de tipo escritorio personal, habiendo elegido instalar GNOME o KDE, requiere al menos 2.1GB de espacio libre. Si selecciona ambos entornos de escritorio, necesitará al menos 1.8GB de espacio libre en disco.

- **Estación de trabajo**

Una instalación de tipo estación de trabajo, incluye un entorno de escritorio gráfico y herramientas de desarrollo de software, requiere al menos 2.1 GB de espacio libre. Si escoge los dos entornos de escritorio GNOME y KDE necesitará al menos 2.4 GB de espacio libre.

- **Servidor**

Una instalación de tipo servidor requiere 850 MB en una instalación mínima sin X-Window (el entorno gráfico), al menos 1.5 GB de espacio libre en disco si todos los componentes que no sean X-Window (grupos de paquetes) están instalados y, al menos, 5.0 GB para instalar todos los paquetes incluidos los entornos GNOME y KDE.

---

<sup>3</sup>El fabricante no ha provisto al mundo linux de la especificacion de su hardware, por lo que hay que usar **sólo** el driver provisto por el fabricante.

<sup>4</sup>Los tamaños son referenciales y pueden variar en una presicion de MB

- **Personalizada**

Una instalación de tipo personalizada requiere 475MB para una instalación mínima y al menos 5.0GB de espacio libre si se selecciona cada uno de los paquetes (TODO).

### 2.3. Arrancar el Instalador

Existen varios métodos que pueden usarse para instalar Red Hat Linux.

Para poder instalar desde un CD-ROM debe disponer de un CD-ROM Fedora core Linux (cd 1) y poseer una unidad de CD-ROM. La mayoría de las computadoras nuevas permitirán arrancar desde el CD-ROM. Si su sistema soporta el arrancar desde el CD-ROM, es fácil empezar una instalación local del CD-ROM.

Al momento de iniciar el sistema, el CD-ROM comenzará a leer la información necesaria para la instalación del Sistema Operativo y aparecerá una pantalla una serie de opciones que permitirán al usuario elegir el método de instalación más apropiado. Desde allí, se podrán acceder a niveles de ayuda que puedan formar la idea de cómo comenzar la instalación del sistema.

### 2.4. Definición del esquema de particiones

El particionamiento<sup>5</sup> le permite dividir el disco duro en secciones aisladas, donde cada sección se comporta como su propio disco duro. El particionamiento es especialmente útil si ejecuta más de un sistema operativo.

Se puede elegir entre realizar un particionamiento automático o un particionamiento manual con Disk Druid. También existe otro particionador llamado `fdisk(8)`, el cual puede ser usado tanto en la instalación como en un sistema funcionando.

Linux utiliza un esquema de nombres que es mucho más flexible y contiene mucha más información que el que usan otros sistemas operativos. Este esquema de nombres está basado en los archivos y tiene la forma:

```
/dev/xxxyN
```

Método para entender el esquema del nombre de la partición:

```
/dev/
```

Esta cadena es el nombre de un directorio en la que están todos los archivos de los dispositivos. Puesto que las particiones residen en el disco y los discos duros son dispositivos, los archivos que representan todas las posibles particiones están contenidos en `/dev/`.

- **xx**

Las dos primeras letras del nombre de la partición se refieren al tipo de periférico en el que se encuentra la misma partición. En general, encontrará `hd` (para discos IDE) o `sd` (para discos SCSI).

- **y**

Esta letra indica en qué dispositivo se encuentra la partición. Por ejemplo, `/dev/hda` (el primer disco duro IDE) o `/dev/sdb` (el segundo disco SCSI).

- **N**

El número que aparece al final indica la partición. Las cuatro primeras (primarias o ampliadas) se

---

<sup>5</sup>Antes de empezar con este procedimiento se recomienda respaldar la información que se encuentra en el disco que será particionado.

enumeran a partir de 1 hasta 4. Las particiones lógicas comienzan en 5. Por ejemplo, `/dev/hda3` es la tercera partición primaria o extendida en el primer disco duro IDE, y `/dev/sdb6` es la segunda partición lógica en el segundo disco SCSI.

Cada una de las particiones creadas tiene atributos físicos y lógicos, de los cuales los más relevantes y de interés para el usuario son; tamaño, tipo de partición, formato, punto de montaje, estos serán discutidos más adelante.

#### 2.4.1. Particionamiento Automatico

El particionamiento automático le permite tener control de los datos que se han eliminado en su sistema. Tiene las siguientes opciones:

**Eliminar todas las particiones Linux del sistema** seleccione esta opción para eliminar tan sólo las particiones Linux (particiones creadas en una instalación Linux previa). No borrará el resto de particiones que tenga en el disco(s) duro(s) (tal como VFAT o particiones FAT32).

**Eliminar todas las particiones del sistema** <sup>6</sup>: seleccione esta opción para eliminar todas las particiones de su disco duro (esto incluye las particiones creadas por otros sistemas operativos tales como Windows 95/98/NT/2000).

**Mantener todas las particiones y usar el espacio libre existente** : Seleccione esta opción para conservar los datos y las particiones actuales, presumiendo que tiene suficiente espacio disponible en los discos duros.

El particionamiento automático creará 3 particiones:

**/boot** : Particion donde se encuentra ubicado el kernel, necesario para el booteo del sistema. Se recomienda que su tamaño no sea inferior a los 100MB.

**swap** : Particion de intercambio. Su tamaño debe ser de al menos 32MB o el doble del tamaño de la ram, el que sea mayor.

**/** : En esta partición se encuentra la raiz del sistema, y su tamaño debe ser apropiado para albergar la instalación que se haya escogido.

Una vez terminada la operación, el instalador preguntara si se desean editar las particiones o continuar. En el primer caso se abrirá *Disk Druid*, cuyo funcionamiento es explicado en la siguiente subsección.

#### 2.4.2. Particionamiento Manual con Disk Druid

Con Disk Druid se pueden crear las particiones de forma manual y de los tamaños que se desee. Disk Druid ofrece una representación gráfica de su/s disco/s duro/s. La información que despliega Disk Druid es la siguiente:

**Dispositivo** : Este campo muestra el nombre del dispositivo de la partición.

---

<sup>6</sup>Esta opción destruirá toda información previa en su disco(s) duro(s)

**Punto de montaje** : Un punto de montaje es el lugar en la jerarquía de directorios a partir del cual un volumen existe; el volumen se "monta" en este lugar. Este campo indica dónde se montará la partición. Si la partición existe pero no se ha definido un punto de montaje, necesitará definir uno.

**Tipo** : Este campo muestra el tipo de partición (por ejemplo, ext2, ext3, o vfat).

**Formato** : Este campo muestra si la partición que se está creando se formateará.

**Tamaño** : Este campo muestra el tamaño de la partición (en MB).

**Comienzo** : Este campo muestra el cilindro en su disco duro donde la partición comienza.

**Final** : Este campo muestra el cilindro en su disco duro donde la partición termina.

*Disk Druid* permite modificar los valores de las particiones existentes, también permite crear nuevas o eliminarlas.

El esquema de particionamiento recomendado por Fedora core es el mismo que se crea la utilizar el particionador automático. Sin embargo existen situaciones en las que se quiera tener un esquema distinto. Cualquier esquema es válido mientras las particiones puedan albergar el sistema instalado. Un ejemplo de esquema sería el siguiente:

Tamaño	Montaje	Dispositivo
100M	/boot	/dev/hde1
30000M	/mnt/xp	/dev/hde3
6000M	/usr	/dev/hde5
800M	/var	/dev/hde6
4000M	/home	/dev/hde7
512M	swap	/dev/hde8
400M	/tmp	/dev/hde9
1000M	/	/dev/hde10
60000M	/opt/mp3	/dev/hde11

Hacer las separaciones mostradas en el ejemplo anterior ayuda a prevenir pérdidas de datos en caso de falla de disco, ya que normalmente sólo se daña una porción de él, por lo que se verían afectadas sólo algunas particiones y no todo el disco. También ayudan a mantener control y orden sobre el sistema.

## 2.5. Preconfigurando el Sistema

El instalador Fedora core nos permite hacer configuraciones en el momento de la instalación, para así poder tener un sistema funcional al momento del primer booteo.

### 2.5.1. Gestor de Arranque

**GRUB** (GRand Unified Bootloader), que se instala por defecto, es un gestor de arranque muy potente ya que puede cargar una gran variedad de sistemas operativos gratuitos así como sistemas operativos de propietarios con el sistema de cargado en cadena (el mecanismo para cargar sistemas operativos no soportados mediante la carga de otro gestor de arranque, tal como DOS o Windows).

### 2.5.1.1. Configuración Básica

Todas las particiones que se pueden arrancar aparecen en una lista, incluso las particiones que usan otros sistemas operativos. La partición que contiene el sistema de ficheros root del sistema tiene la Etiqueta de Fedora core para GRUB. Las otras particiones puede que también tengan etiquetas de arranque. Si desea añadir o cambiar la etiqueta de arranque de las otras particiones que el programa de instalación ya ha detectado, selecciónela y modifique.

Seleccione Por defecto junto con la partición root preferida para escoger el sistema operativo que se desee arrancar por defecto. No podrá avanzar en la instalación mientras no escoja la imagen de arranque por defecto.

Las contraseñas del gestor de arranque ofrecen un mecanismo de seguridad en un ambiente en el que se tenga acceso físico al servidor.

Si está instalando un gestor de arranque, debe crear una contraseña para proteger el sistema. Sin dicha contraseña, los usuarios con acceso a su sistema pueden pasar opciones al kernel que pueden poner en compromiso la seguridad de su sistema. Con la contraseña, se tiene que introducir para poder seleccionar cualquier opción de arranque que no sea estándar.

Si selecciona colocar una contraseña para aumentar la seguridad del sistema, asegúrese de seleccionar la casilla **Usar la contraseña del gestor de arranque**.

Una vez seleccionada, introduzca la contraseña y confírmela.

### 2.5.1.2. Configuración Avanzada

Ahora que ha decidido cuál gestor de arranque instalar, tiene que decidir dónde quiere instalarlo. Puede instalar el gestor de arranque en uno de los dos sitios siguiente:

- El master boot record (MBR)  
Este es el sitio recomendado para instalar un gestor de arranque, a no ser que el MBR esté ocupado por el gestor de arranque de otro sistema operativo, como System Commander. El MBR es un área especial del disco duro que la BIOS de su computadora carga automáticamente y el punto más próximo en el que el gestor de arranque puede tomar el control de la secuencia de arranque. Si lo instala en el MBR, al arrancar su máquina, GRUB presentará un indicador de comandos de arranque. Podrá entonces iniciar Fedora core o cualquier otro sistema operativo que le haya indicado al gestor de arranque.
- El primer sector de la partición raíz  
Se recomienda si está utilizando otro gestor de arranque en su sistema. En este caso, el otro gestor de arranque tendrá el control en un primer momento. Podrá configurar ese gestor de arranque para que inicie GRUB, que iniciará a su vez Fedora core.

Si el sistema sólo utilizará Fedora core, debería seleccionar el MBR. Para sistemas con Windows 95/98, también debería instalar el gestor de arranque en el MBR para que se puedan iniciar los dos sistemas operativos.

La opción **Forzar el uso de LBA32** (no requerida normalmente) le permite exceder el límite de cilindro 1024 para la partición /boot. Si posee un sistema que es compatible con la extensión LBA32 para arrancar los sistemas operativos por encima del límite de cilindro 1024 y desea ubicar la partición /boot más allá de este límite, debería seleccionar esta opción.

### 2.5.2. Configuración de red

Si tiene un dispositivo de red y no ha configurado todavía su red (como por ejemplo proporcionar un disco de arranque de red que haya creado y entrar en la información de red como se indica), tiene la oportunidad de hacerlo.

El programa de instalación automáticamente detecta los dispositivos de red que tiene y los muestra en la lista Dispositivos de red.

Una vez que haya seleccionado el dispositivo de red, lo podrá modificar. En la pantalla desplegable Modificar interfaz puede elegir la dirección IP o la máscara de red del dispositivo con el DHCP (o manualmente si no ha seleccionado DHCP ) y puede también activar el dispositivo en el intervalo de arranque. Si selecciona Activar en arranque, el dispositivo de red arrancará cuando arranque el sistema. Si no tiene el acceso al cliente DHCP o no está seguro contacte con el administrador de red.

### 2.5.3. Configuración del cortafuegos

Fedora core Linux también le ofrece protección vía cortafuegos (firewall) para una seguridad mejorada del sistema. Un cortafuegos se dispone entre su ordenador y la red y determina qué recursos de su equipo están accesibles para los usuarios remotos de la red. Un cortafuegos bien configurado puede aumentar significativamente la seguridad de su sistema.

Seleccione el nivel de seguridad apropiado del sistema.

**Alto** Si elige Alto, su sistema no aceptará más que las conexiones (además de las definidas por defecto) que hayan sido explícitamente definidas por usted.

**Medio** Si elige Medio, su cortafuegos no permitirá a las máquinas remotas acceder a ciertos recursos de su sistema.

**Ningún cortafuegos** La configuración "ningún cortafuegos" proporciona un acceso completo al sistema y no realiza ningún tipo de verificación de seguridad. La comprobación de seguridad es la desactivación del acceso a determinados servicios. Tan sólo se recomienda esta opción si está usando una red certificada y segura (no Internet), o si planea realizar una configuración detallada del cortafuegos más adelante.

### 2.5.4. Selección del soporte del idioma

Puede instalar y soportar múltiples idiomas para usar en su sistema.

Debe instalar un idioma para usarlo como idioma por defecto. El idioma por defecto será usado por el sistema una vez que la instalación se haya completado. Si escoge instalar otros idiomas, puede cambiar su idioma por defecto tras la instalación.

Si tan sólo va a utilizar un idioma en su sistema, podrá ganar bastante espacio en disco. El idioma por defecto es el idioma que haya seleccionado durante el proceso de instalación.

### 2.5.5. Configuración del huso horario

Puede elegir su huso horario o bien seleccionando la localización física de su ordenador o bien especificando su huso horario en función del Universal Time Coordinated (UTC).

### 2.5.6. Configuración de la contraseña de *root*

La configuración de la cuenta y la contraseña *root* es uno de los pasos más importantes durante la instalación. La cuenta *root* es usada para instalar paquetes, actualizar RPMs y realizar la mayoría de las tareas de mantenimiento del sistema. Conectándose como *root* le dá control completo sobre el sistema.

El programa de instalación le dará indicaciones para que configure una contraseña de *root* para su sistema. Debe introducir una contraseña de *root*. El programa de instalación no le permitirá que pase a la siguiente sección sin introducir una contraseña de *root*.

La contraseña de *root* debe de tener al menos seis caracteres y no aparecerá en la pantalla cuando la teclee. Deberá introducirla dos veces; si las dos contraseñas no coinciden, el programa de instalación le pedirá que las vuelva a introducir.

Debería escribir una contraseña de *root* fácil de recordar, pero que no sea obvia o fácil de adivinar. Su nombre, su número de teléfono, *qwerty*, *contraseña*, *root*, *123456* y *anteayer* serían ejemplos de malas contraseñas. Las contraseñas mejores son aquéllas que mezclan números con letras mayúsculas y minúsculas que no formen palabras contenidas en diccionarios, como por ejemplo : *Aard387vark* o *420BMttNT*. Recuerde que la contraseña es sensible a las mayúsculas y minúsculas. Se recomienda que nunca escriba su contraseña pero, si la escribe en un papel, guárdelo en un lugar seguro.

### 2.5.7. Configuración de la autenticación

Puede saltarse esta sección si no va a configurar contraseñas de red. Si no sabe por qué debería hacer esto, contacte con su administrador de sistemas.

A no ser que esté utilizando autenticación *NIS* o *LDAP*, verá que sólo las contraseñas tipo MD5 y shadow están seleccionadas. Le recomendamos que utilice ambos tipos de contraseñas para que su máquina sea lo más segura posible.

**Habilitar contraseñas MD5:** le permite usar una contraseña larga (de hasta 256 caracteres), en vez de las ocho letras o menos estándar.

**Habilitar contraseñas shadow:** proporciona un método seguro para conservar contraseñas. Las contraseñas se almacenan en */etc/shadow*, al que tan sólo se puede acceder como *root*.

Existen más opciones de autenticación, pero su discusión va más allá de los alcances de este documento.

## 2.6. Selección de paquetes

Anteriormente se discutió que existían distintos tipos de instalaciones. El instalador dará la opción de modificar el conjunto de paquetes que auto seleccionó basándose en si quería una instalación de escritorio o un servidor. Se podrán seleccionar grupos de paquetes y paquetes individuales para hacer amoldar la instalación a lo que uno necesita.

### 2.6.1. Selección individual de paquetes

Tras haber seleccionado los paquetes que quiera instalar, podrá seleccionar o anular la selección de los paquetes individualmente.

Puede escoger visualizar los paquetes individuales en Vista de árbol o Vista plana.

La Vista de árbol le permite ver los paquetes agrupados según el tipo de aplicación.

La Vista plana le permite ver todos los paquetes listados en orden alfabético en la parte derecha de la pantalla.

### 2.6.2. Dependencias no satisfechas

Una vez seleccionado y prosiguiendo con la instalación, puede que el instalador abra un diálogo en el que pregunta que hacer con los paquetes cuyas dependencias no estan satisfechas. En este caso se podra instalar de todas maneras<sup>7</sup>, obviar la instalacion de los paquetes conflictivos o satisfacer sus dependecias instalando paquetes adicionales..

## 2.7. Finalizando la instalación

El programa de instalación a continuación le proporcionará una lista de tarjetas de vídeo entre las que escoger.

Si decide instalar los paquetes del Sistema X Window, tendrá la oportunidad de configurar un servidor X para su sistema.

### 2.7.1. Tarjeta de Video

Si su tarjeta de vídeo no aparece en la lista, X puede que no la soporte. No obstante, si posee conocimiento técnico sobre su tarjeta, puede escoger Tarjeta no listada e intentar configurarla al hacer corresponder su chipset de tarjeta de vídeo con uno de los servidores X disponibles.

### 2.7.2. Monitores

El programa de instalación le presentará una lista de monitores de la que seleccionar. Desde esta lista, puede usar el monitor que se detecte de forma automática o escoger otro monitor.

Si está instalando el sistema Fedora core en un portátil con una pantalla LCD, deberá seleccionar el modelo Genérico más adecuado.

Si su monitor no aparece en la lista, seleccione el modelo Genérico más apropiado dentro de los modelos disponibles. Si selecciona un monitor Genérico, el programa de instalación le sugerirá valores de sincronización horizontales y verticales. Estos valores suelen encontrarse en la documentación que acompaña al monitor o los puede obtener a través del vendedor o fabricante del monitor; compruebe la documentación para asegurarse de que estos valores se han establecido correctamente.

### 2.7.3. Configuración de X-Window

Elija la densidad del color y la resolución para su configuración de X.

Si está realizando una instalación personalizada o de servidor, también puede escoger si desea arrancar su sistema en modo gráfico o texto una vez que la instalación se termine. A menos que tenga necesidades especiales, se recomienda el arranque en ambiente gráfico (similar al entorno Windows). Si elige arrancar en un ambiente de texto, se le presentará una línea de comandos (similar al entorno DOS).

---

<sup>7</sup>Podría causar que algunos programas no funcionen correctamente

Las instalaciones de escritorio personal y de estación de trabajo automáticamente arrancarán en ambiente gráfico.

## 2.8. Tareas posteriores a la instalación

Ya teniendo un sistema con una instalación fresca, será necesario hacer algunos últimos ajustes para que la computadora pueda entrar en producción.

Es de vital importancia el actualizar el sistema una vez instalado. Normalmente las actualizaciones existentes corrijen fallas de seguridad en las distintas aplicaciones del sistema.

Ahora se pueden reconfigurar todos los dispositivos si es necesario. Para ello se pueden utilizar las herramientas `system-config-ALGO`. Estas herramientas permiten reconfigurar las opciones escogidas durante la instalación. También se pueden hacer las modificaciones de forma manual, como se ha visto en secciones anteriores.

## 3. Configuración de dispositivos

Uno de los temas por lo que más gente le tiene miedo a Linux, es el hecho de tener que configurar sus dispositivos de manera no tan automática. Por ello, las últimas versiones de las distribuciones Linux traen incorporadas una serie de herramientas, que permiten al usuario instalar y configurar de manera mucho más fácil y amigable sus dispositivos.

Esto tiene también su salvedad, ya que necesitamos que el dispositivo sea soportado por cualquiera de estas herramientas de configuración, por lo tanto, necesitaremos estar bastante seguros de qué es lo que necesitamos instalar (no instalaremos cualquier cosa porque sí), y también tenemos que saber las características del dispositivo que queremos instalar; como su modelo, chipset y algunas características que detallaremos más adelante.

La gran ventaja que presenta Fedora core es que pueden configurarse los distintos dispositivos, con las herramientas que provee la distribución `system-config-ALGO`, donde `ALGO` puede ser mouse, network, soundcard, display etc.

Se debe tener en cuenta también, que en algunos casos como el Teclado o mouse, la instalación de algún dispositivo va a diferir si es para hacerlo funcionar en el entorno gráfico o en la consola, por lo que se deberá configurar distintos archivos para tal efecto.

### 3.1. Teclado

Algo que ocurre con mucha frecuencia entre los nuevos usuarios de Linux, es que debido a una distracción durante el procesos de instalación, encontrarse con un mapa de teclado en inglés. Hay métodos muy sencillos que permitirán establecer el mapa de teclado correcto.

Una vez configurado el mapa del teclado, algunos usuarios pueden encontrarse atónitos al intentar, desde el modo gráfico, encontrar como desplegar una simple @ en una dirección de correo electrónico, así como otros caracteres.

Este procedimiento puede hacerse de dos métodos: desde una terminal o consola y `system-config-keyboard`. El primero es el método más efectivo, y cierta forma el más complicado, se logra editando el archivo `/etc/X11/xorg.conf`, lo cual determinará el mapa del teclado para el entorno gráfico, y se obtiene modificando el valor de la variable `XkbLayout` en la sección `InputDevice`

```
XkbLayout      "es"
```

Por otro lado, en el entorno consola, se debe editar `/etc/sysconfig/i18n` y cambiar las incidencias del idioma incorrecto por el que corresponda a su propio idioma y región:

```
LANG="es_CL.ISO8859-1"
SUPPORTED="es_CL.ISO8859-1:en_US:en:es_ES.UTF-8:es_ES:es"
SYSFONT="latarcyrheb-sun16"
```

Finalmente se debe editar `/etc/sysconfig/keyboard` y cambiar las incidencias del mapa de teclado incorrecto, por el que corresponda a su idioma y región:

```
KEYBOARDTYPE="pc"
KEYTABLE="es"
```

Otra forma de manejar esto, es utilizando la herramienta `system-config-keyboard`, que hará lo propio para el modo terminal o consola, y especificando el mapa de teclado deseado.

### 3.2. Mouse

De manera similar al teclado (y como casi todo en Linux), el mouse puede ser configurado a través de la edición de algún archivo o con la herramienta (`system-config-mouse`)

Al Editar el archivo `/etc/sysconfig/mouse` debemos asegurar de que contenga el tipo de mouse correcto:

```
FULLNAME="Generic - 3 Button Mouse (PS/2)"
MOUSETYPE="ps/2"
XEMU3="no"
XMOUSETYPE="PS/2"
DEVICE=/dev/input/mice"
```

**Nota:** Si el mouse que está ocupando tiene la ruedita de scroll, utilice IMPS/2 en vez de PS/2.

A continuación, se puede editar el archivo `/etc/X11/xorg.conf` y en la sección `"InputDevice"` se puede tener lo siguiente:

```
Section "InputDevice"
    Identifier "Mouse0"
    Driver      "mouse"
    Option      "Device" "/dev/input/mice"
    Option      "Protocol" "PS/2"
    Option      "Emulate3Buttons" "off"
    Option      "ZAxisMapping" "4 5"
EndSection
```

Donde nuevamente se puede reemplazar PS/2 por IMPS/2 para obtener el funcionamiento del scroll.

### 3.3. Video

El soporte para las tarjetas de Video lo provee el conjunto de paquetes *X.org*, el cual viene con la distribución.

Primero que todo, debemos asegurarnos que disponemos del hardware apropiado para ejecutar el sistema X Window, la cantidad de memoria adecuada y el espacio de disco necesario.

Son necesarios unos 150 a 200 MB de espacio en disco para instalar el sistema XFree86 junto con las aplicaciones suministradas. Necesita al menos de 16MB de memoria virtual para ejecutar *X.org*.

La **memoria virtual** es la combinación de la RAM física en su sistema y de la cantidad de espacio swap que haya reservado a Linux. Debe tener al menos 4 MB de RAM física para ejecutar bajo Linux, por lo que requerirá un archivo swap de 12MB. Cuanta más RAM física tenga, mejor rendimiento obtendrá su sistema *X.org*.

#### 3.3.1. Instalando X.org

*X.org* está dividido en un gran número de paquetes RPM. Algunos son obligatorios, pero otros opcionales. Si instaló *X.org* en el proceso de instalación del sistema, probablemente ya ejecutó automáticamente el proceso. Si no lo hizo, entonces debemos hacerlo ahora.

Los paquetes de *X.org* recomendados son:

RPM	Descripción
xorg-x11	Sistema xorg base.
xorg-tools	Muchas aplicaciones X útiles
xorg-x11-xfs	Servidor de fuentes standard X
xorg-x11-font-utils	Paquetes para instalar fuentes
fonts-xorg-base	Fuentes standard X
xorg-x11-libs	Bibliotecas compartidas para la mayoría de las aplicaciones X
xorg-x11-doc	Documentacion X.org

Una vez que tenga los paquetes RPM, lo normal es que pueda instalarlos fácilmente usando rpm

```
> rpm -ivh *xorg*.rpm
```

#### 3.3.2. Configurando X.org

Derivado de su predecesor XFree86, históricamente *X.org* ha sido una de las partes más complejas bajo Linux, en lo que respecta a su configuración. Este ya no es el caso para el hardware más habitual. Sin embargo, aún hay dos casos en los que la instalación puede ser dificultosa.

En primer lugar, el Hardware de más reciente aparición puede estar soportado por *X.org*, o puede no estarlo en absoluto. Si *X.org* lo soporta, puede que tenga que usar versiones beta de *X.org*, o incluso versiones parchadas del mismo. Éstas no estarán soportadas por las nuevas herramientas de configuración.

En segundo lugar, algunos proveedores no publican las especificaciones para sus tarjetas. Para que *X.org* soporte estas tarjetas, los desarrolladores deben efectuar una ingeniería inversa, lo que lleva mucho tiempo y esfuerzos. A menos que la tarjeta sea de uso extendidísimo, puede que no haya soporte de xorg durante mucho tiempo. Hacer que una tarjeta sin soporte funcione puede resultarle difícil, aunque no imposible, si no es capaz de escribir usted mismo los controladores.

Éstos son sólo los peores casos. Para Hardware más extendido, le será suficiente usar *system-config-*

*xfree86* y debería funcionar todo sin mayor problema.

*X.org* tiene un archivo de configuración en `/etc/X11/xorg.conf`. Este archivo está dispuesto en secciones con el siguiente formato:

```
Section "Nombre de la seccion"
  Comando1 "Opción"
  Comando2 "Opción"
  Subsection "Nombre de la subsección"
    Comando3 "Opción"
  EndSubSection
EndSection
```

Las secciones que podemos encontrar son:

- Modules
- Files
- ServerFlags
- InputDevice
- Monitor
- Device
- Screen
- ServerLayout

y cada una de estas secciones contendrá opciones que podrán ser “seteadas” para describir qué es lo que se quiere.

La más común de configurar es la que tiene que ver con el monitor y con la tarjeta de video.

En la sección `Screen` se encuentra información referente al modo de pantalla que se ejecutará al momento de iniciar un servicio de *X*. Allí se puede ver dentro de las subsecciones `Display`, los parámetros para configurar la resolución (`Modes`) y profundidad de colores (`Depth`) que tendrá el monitor.

Si tenemos más de una subsección `Display`, podremos cambiarla una vez iniciado nuestro servidor *X* con las teclas `CTRL ALT` y el signo `+ o -` según corresponda. Por ejemplo, se puede tener algo como esto:

```
Section "Screen"
  Identifier "Screen0"
  Device      "Videocard0"
  Monitor     "Monitor0"
  DefaultDepth 24
  SubSection "Display"
    Viewport 0 0
    Depth 16
    Modes "800x600"
  EndSubSection
```

```
SubSection "Display"
    Viewport 0 0
    Depth 24
    Modes "800x600"
EndSubSection
EndSection
```

Por otro lado, la sección ``Monitor`` corresponde a la definición del monitor que se está ocupando:

```
Section "Monitor"
    Identifier "Monitor0"
    VendorName "Monitor Vendor"
    ModelName "Olidata MR 1502"
    HorizSync 30.0 - 50.0
    VertRefresh 50.0 - 120.0
    Option "dpms"
EndSection
```

Donde generalmente la parte más difícil de encontrar es la referente a las tasas de refresco horizontal y vertical (`HorizSync` y `VertRefresh`), las cuales pueden ser encontradas en Internet a través de Google<sup>8</sup>, o en la página web del proveedor del monitor.

Por último, queda por configurar la sección que hace referencia a la tarjeta de video que se está utilizando. Para ello, se debe hacer referencia a la sección ``Device``, la cual tiene una nomenclatura similar a la siguiente:

```
Section "Device"
    Identifier "Videocard0"
    Driver "savage"
    VendorName "Videocard vendor"
    BoardName "S3 ProSavage KM133"
EndSection
```

El driver (ubicado en `/usr/X11R6/lib/modules/drivers/`), es el encargado de *traducir* lo que dice la tarjeta de video al sistema y representarlo a través del monitor.

### 3.3.3. nVidia

Para aquellas personas que posean tarjetas de video con aceleración gráfica y ocupen el chipset nVidia, se sugiere instalar el driver que provee esta misma compañía, en vez del que provee Fedora core. Para hacer esto, uno puede ingresar a la página de nVidia en <http://www.nvidia.com/linux> y bajar el último driver que provean ellos (por ejemplo la versión 1.0-6629) y ejecutar el archivo bajado de la siguiente manera (se asume que el usuario que ejecuta esto es el root y las fuentes del kernel están instaladas en la máquina):

---

<sup>8</sup><http://www.google.com/linux>

```
sh NVIDIA-Linux-x86-1.0-6629-pkg1.run
```

Luego, se procede a comentar las secciones del archivo `/etc/X11/xorg.conf` que hagan relación al `dri` y al `glcore`. A continuación se carga el modulo de `nvidia`

```
modprobe nvidia
```

y se copia el driver de `nvidia` a un nuevo directorio, para que pueda ser reconocido por `udev`, con dueño y grupo perteneciente a `root`:

```
cp -a /dev/nvidia* /etc/udev/devices
chown root.root /etc/udev/devices/nvidia*
```

Luego de reiniciar el sistema `X`, se podrá ver que al momento de levantar la aplicación, muestra una imagen (o splash) de `nVidia`

Adicionalmente, si se quiere sacar un mayor provecho a las extensiones que provee la última versión de `X.org`, se puede agregar lo siguiente a `/etc/X11/xorg.conf`

```
Section "Extensions"
    Option "Composite" "Enable"
EndSection
```

... aunque cabe destacar que esta opción también funcionará sin el driver `nVidia`, pero muchísimo más lento.

### 3.4. Sonido

La configuración de este dispositivo depende mucho de la tarjeta de sonido que esté instalada. Para ello, se necesitará cargar el módulo apropiado, dependiendo del chip de la tarjeta. Supongamos que tenemos una tarjeta de sonido `PCI`, y queremos obtener información de ésta. Con el comando `lspci -v` nos arrojará los nombres de todos los dispositivos `PCI` que estén conectados en nuestro computador.

Luego, una vez conocido el nombre del dispositivo, podemos agregar el módulo correspondiente a `/etc/modules.conf`

La manera simple de configurar la tarjeta de sonido, basta con escribir el comando `system-config-soundcard` en el intérprete del shell para lanzar la Herramienta de configuración de la tarjeta de sonido. Si no es `root`, le pedirá su contraseña de superusuario para continuar.

### 3.5. Adaptadores de red

Configurar los parámetros de red en una estación de trabajo `Linux` o un servidor no es realmente complicado. Solamente requiere de algunos conocimientos básicos sobre redes y cualquier editor de texto plano.

La marca de la tarjeta, es lo que menos interesa, lo que es importante es que se determine con exactitud que chipset utiliza ésta. Esto puede determinarse examinando físicamente la tarjeta de red o bien examinando a detalle la salida en pantalla que se obtiene al ejecutar el siguiente comando:

```
lspci | grep Ethernet
```

Lo cual devuelve una salida similar a las siguiente (en el caso de una tarjeta 3Com 905 C)

```
Ethernet controller: 3Com Corporation 3c905C-TX [Fast Etherlink] (rev 120).
```

Debe entonces editarse `/etc/modprobe.conf` y verificar que el módulo de la tarjeta de red, esté especificado correctamente. Ejemplo:

```
alias eth0 3c59x
```

Si se realizó alguna edición de este fichero, deberá de ejecutarse el siguiente comando, a fin de actualizar dependencias:

```
/sbin/depmo -a
```

Como Fedora core 3 utiliza kernel 2.6.x, la lista de módulos existentes en el equipo que puede utilizar para distintos chipsets, de diferentes tarjetas de red se puede obtener listando el contenido del directorio `/lib/modules/[versión de su kernel]/kernel/drivers/net/`. Ejemplo:

```
ls /lib/modules/2.6.9-1.681_FC3/kernel/drivers/net/
```

Debe editarse con un procesador de textos `/etc/sysconfig/network` y en este establece la puerta de enlace (Gateway) y su nombre de máquina. Ejemplo:

```
NETWORKING=yes  
HOSTNAME=nombre_maquina.nombre_dominio.cl  
GATEWAY=192.168.1.254
```

Luego editar `/etc/sysconfig/network-scripts/ifcfg-eth0` y verificar que sus parámetros de red sean los correctos. Ejemplo:

```
DEVICE=eth0  
BOOTPROTO=static  
IPADDR=192.168.1.50  
NETMASK=255.255.255.0  
NETWORK=192.168.1.0  
BROADCAST=192.168.1.255  
ONBOOT=yes
```

Los parámetros anteriores son proporcionados por el administrador de la red local en donde se localice la máquina que está siendo configurada. El administrador de la red deberá proporcionar una dirección IP (IPADDR), una máscara de la subred (NETMASK), dirección IP de la red (NETWORK) y el Broadcast (BROADCAST). Si por otro lado, el administrador de la red, le dice que la configuración se realizará de manera automática mediante **DHCP**, entonces se debe dejar el archivo de la siguiente manera:

```
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
```

Debe editarse con un procesador de textos */etc/hosts*, y debe verificarse que este diferenciado el loopback del nombre de la máquina. Ejemplo:

```
192.168.1.50 su_maquina.su_dominio.com su_maquina
127.0.0.1 localhost.localdomain localhost
```

Y finalmente, debe editar */etc/resolv.conf* y establecerse los servidores de resolución de nombres de dominio (DNS). Ejemplo:

```
nameserver 192.168.1.254
nameserver 192.168.1.1
```

Después de hacer todo lo anterior, solo deberá de ser reiniciado el servicio de red. Debe ejecutarse el siguiente comando:

```
service network restart
```

Basta solamente comprobar si hay realmente conectividad. Puede ejecutarse el comando ping hacia cualquier dirección de la red local para tal fin.

```
ping 192.168.1.254
```

Obviamente, toda esta configuración también se puede hacer mediante la herramienta *system-config-network*

### 3.6. Impresoras

Desde la versión Red Hat 9, *CUPS* es el sistema de impresión predeterminado. Sin embargo, todavía se proporciona el sistema de impresión por defecto anterior, *LPRng*. Si el sistema fue actualizado desde una versión anterior de Red Hat o Fedora core, que usaba *LPRng*, el proceso de actualización no reemplaza *LPRng* con *CUPS*; el sistema continuará usando *LPRng*.

Si un sistema fue actualizado desde una versión anterior de Red Hat o Fedora core que usaba *CUPS*, el proceso de actualización mantiene las colas de impresión configuradas y el sistema continuará usando *CUPS*.

La forma más simple de configurar la impresora, es utilizando la herramienta *system-config-printer*, y se puede forzar a la Herramienta de configuración de impresoras a ejecutarse como una aplicación basada en texto usando el comando *system-config-printer-tui* desde el intérprete de comandos.

### 3.7. Tuning del sistema

El comando *hdparm* de Linux nos permite optimizar al máximo nuestras unidades IDE/UDMA, aunque con cierto riesgo. Las distribuciones de Linux suelen suponer que nuestro equipo es muy antiguo

y no tiene estas características técnicas de las que tanto presumen los fabricantes de discos duros. Y lo hacen por seguridad, para garantizar que el acceso a los datos funcione siempre bien, sin errores, ni corrupción de datos. Si forzamos el uso de los modos PIO, o el UltraDMA, empezamos a correr ciertos riesgos, aunque por otro lado, no reconoceremos nuestro propio sistema al lanzar las X, dado que se mostrarán en pantalla de inmediato.

Lo primero que haremos es averiguar los discos duros y particiones que tiene Linux en nuestro equipo. Veamos un ejemplo:

```
# dmesg | grep hd
hda: SAMSUNG VG36483A (6.48GB), ATA DISK drive
hdb: ST340823A, ATA DISK drive
hda: 12685680 sectors (6495 MB) w/494KiB Cache, CHS=789/255/63
hdb: 78165360 sectors (40021 MB) w/512KiB Cache, CHS=4865/255/63
hda: hda1 hda2
hdb: hdb1
```

Aquí vemos 2 discos: hda y hdb. Vamos a optimizar el primero, hda. Para ello, sin precipitarnos, extraemos cierta información del disco, que puede ayudarnos a buscar la mejor configuración para hdparm:

```
# /sbin/hdparm -gi /dev/hda
/dev/hda:
geometry   = 789/255/63, sectors = 12685680, start = 0

Model=SAMSUNG VG36483A (6.48GB), FwRev=FL100, SerialNo=TS840703352e3f
Config={ HardSect NotMFM HdSw>15uSec Fixed DTR>10Mbps }
RawCHS=13424/15/63, TrkSize=32256, SectSize=512, ECCbytes=21
BuffType=3(DualPortCache), BuffSize=494kB, MaxMultSect=16, MultSect=16
DblWordIO=no, OldPIO=2, DMA=yes, OldDMA=2
CurCHS=13424/15/63, CurSects=-1854930751, LBA=yes, LBASects=12685680
tDMA={min:120,rec:120}, DMA modes: sword0 sword1 sword2 *mword0 mword1 mword2
IORDY=on/off, tPIO={min:120,w/IORDY:120}, PIO modes: mode3 mode4
UDMA modes: mode0 model *mode2
```

Es conveniente tomar nota de la información extraída, para su uso posterior. Antes de optimizar, vamos a realizar un test de velocidad, para luego poder comparar.

```
# /sbin/hdparm -t -T /dev/hda
/dev/hda:
Timing buffer-cache reads: 128 MB in 2.18 seconds = 58.72 MB/sec
Timing buffered disk reads: 64 MB in 11.89 seconds = 5.38 MB/sec
```

La opción -T realiza un test del sistema de cache (o sea, la memoria, la CPU, y el cache de buffer). Por contra, la opción -t mide el acceso al disco sin usar el cache. Estas dos medidas nos pueden dar una idea del rendimiento de I/O del disco.

El estado actual de configuración de la unidad de disco nos lo muestra el siguiente comando:

```
# /sbin/hdparm /dev/hda
/dev/hda:
```

```
multcount = 0 (off)
I/O support = 0 (default 16-bit)
unmaskirq = 0 (off)
using_dma = 0 (off)
keepsettings = 0 (off)
nowerr = 0 (off)
readonly = 0 (off)
readahead = 8 (on)
geometry = 789/255/63, sectors = 12685680, start = 0
```

Al parecer nuestro disco no está usando ninguna de las características potentes que dispone: modo 32 bits, UDMA, modos PIO, etc.

Pasamos nuestro sistema Linux a modo monousuario y ejecutamos el siguiente comando:

```
# /sbin/hdparm -X66 -u1 -m16 -c3 -W1 /dev/hda
/dev/hda:
setting 32-bit I/O support flag to 3
setting multcount to 16
setting unmaskirq to 1 (on)
setting xfermode to 66 (UltraDMA mode2)
setting drive write-caching to 1 (on)
multcount = 16 (on)
I/O support = 3 (32-bit w/sync)
unmaskirq = 1 (on)
```

Esto se ve mucho mejor, veamos cuán rápido accede ahora a los datos el disco duro:

```
# /sbin/hdparm -t -T /dev/hda
/dev/hda:
Timing buffer-cache reads: 128 MB in 2.31 seconds = 55.41 MB/sec
Timing buffered disk reads: 64 MB in 7.48 seconds = 8.56 MB/sec
```

Parece que el rendimiento de acceso directo se ha incrementado en más de un 50%. Dependiendo del modelo de disco y configuración hardware de nuestro equipo, podemos incrementar este valor hasta en un 1.000%, por lo que bien puede merecer la pena, en general.

La configuración que establecemos con el comando `/sbin/hdparm` se pierde al reiniciar Linux, por lo que conviene definirla en los scripts de arranque, por ejemplo en `/etc/rc.d/rc.sysinit`:

```
# Se optimiza al maximo el acceso al disco IDE: UDMA, PIO, etc
/sbin/hdparm -X66 -u1 -m16 -c3 -W1 /dev/hda

# Se fuerza un chequeo completo de disco al arrancar
e2fsck /dev/hda1 -f -p -y

if [ -f /fsckoptions ]; then
    fsckoptions='cat /fsckoptions'
else
    fsckoptions=
fi
```

Quizá, en algún caso, la distribución que tenemos instalada se haya tomado la libertad de configurar por nosotros esta opción. Para averiguarlo, podemos intentar buscar una llamada a `hdparm` en los scripts de arranque del siguiente modo:

```
# grep hdparm /etc/* -r
```

`hdparm` tiene muchas opciones, y los discos duros nuevos traen muchas características técnicas interesantes. Encontrar la configuración más optimizada no es fácil. Se recomienda por lo tanto, revisar con sumo cuidado el manual de éste.

## 4. Trabajando como root

### 4.1. Precauciones

El usuario `root` es el dueño y señor de la máquina en la cual uno está trabajando. Tener acceso a utilizar la cuenta del superusuario o `root` implica una gran responsabilidad, por lo que se sugieren las siguientes precauciones:

- No modifique nada que no pueda volver a dejar como estaba. En otras palabras, **siempre** antes de hacer algo asegúrese la posibilidad de poder retroceder fácilmente a la situación inicial.
- Maneje el password de `root` sólo ud. y nadie más que ud. Es una muy mala idea compartir cualquier tipo de claves. Además la administración de un sistema debe llevarla sólo una persona para mantener orden y evitar conflictos.
- No entrar a X como `root`, genera vulnerabilidades tanto por el dueño del usuario, como potenciales ataques de agentes externos.
- No modificar parámetros si no conoce la función de qué es lo que realmente hace (No jugar a *conocer cosas como root*).
- No correr servicios como `root`, ya que éstos pueden estar expuestos a vulnerabilidades que pueden ser explotadas para utilizar la máquina en contra de uno.

### 4.2. Trabajando con servicios

El sistema de niveles de ejecución `SysV init` provee de un proceso estándar para controlar cuáles programas `init` lanza o detiene cuando se inicializa un nivel de ejecución. `SysV init` fué escogido porque es más fácil de usar y más flexible que el proceso tradicional `init` estilo BSD.

Los ficheros de configuración para `SysV init` están en el directorio `/etc/rc.d/`. Dentro de este directorio, se encuentran los scripts `rc`, `rc.local`, `rc.sysinit`, y, opcionalmente, los *scripts* `rc.serial` así como los siguientes directorios:

```
init.d/  
rc0.d/  
rc1.d/  
rc2.d/  
rc3.d/
```

```
rc4.d/  
rc5.d/  
rc6.d/
```

El directorio `init.d/` contiene los scripts usados por el comando `/sbin/init` cuando se controlan los servicios. Cada uno de los directorios numerados representa los seis niveles de ejecución predeterminados configurados por defecto bajo Fedora core.

#### 4.2.1. Niveles de ejecución

Los niveles de ejecución son un estado, o modo, definido por los servicios listados en el directorio `/etc/rc.d/rc<x>.d/`, donde `ix` es el número de nivel de ejecución.

La idea detrás de los niveles de ejecución de *SysV init* gira alrededor del hecho que sistemas diferentes se pueden usar de formas diferentes. Por ejemplo, el servidor corre de forma más eficiente sin tener que arrastrar recursos del sistema creados por el sistema X. Otras veces, el administrador del sistema puede necesitar operar el sistema en un nivel más bajo de ejecución para realizar tareas de diagnóstico, como reparar corrupción del disco duro, cuando no es posible que ningún otro usuario esté usando el sistema.

Las características de un nivel de ejecución dado, determinan qué servicios son detenidos o iniciados por `init`. Por ejemplo, el nivel de ejecución 1 (modo único usuario) detiene cualquier servicio de red, mientras que el nivel 3 arranca estos servicios. Asignando servicios específicos a ser detenidos o arrancados en un nivel dado, `init` puede fácilmente cambiar el modo de la máquina sin que el usuario tenga que manualmente arrancar o detener servicios.

Los siguientes niveles de ejecución están definidos por defecto para Fedora core:

- 0 - Halt (detener)
- 1 - Modo mono-usuario
- 2 - Modo Multiusuario, sin NFS(network file system)
- 3 - Modo Full multiusuario
- 4 - Sin uso
- 5 - Nivel 3 + Ambiente Gráfico(X11)
- 6 - Reboot (reiniciar)

Generalmente, los usuarios utilizan Fedora core en el nivel de ejecución 3 o nivel de ejecución 5: ambos modos multiusuario. Ya que los niveles de ejecución 2 y 4 no son usados, los usuarios a veces personalizan estos niveles para cubrir necesidades específicas.

El nivel de ejecución por defecto para el sistema está listado en `/etc/inittab`. Para saber el nivel de ejecución por defecto de un sistema, busque por la línea similar a la que se muestra abajo cerca de la parte superior de `/etc/inittab`:

```
id:5:initdefault:
```

El nivel de ejecución predeterminado en el ejemplo de arriba es cinco, como indica el número después del identificador `id`. Para cambiarlo, modifique `/etc/inittab` como usuario `root`.

Tenga mucho cuidado cuando esté modificando `/etc/inittab`. Errores simples de tipeo pueden hacer que su sistema no arranque nuevamente. Si esto ocurre, use un disquete de arranque, entre a *modo de usuario único* o entre en *modo de rescate* y repare el archivo.

#### 4.2.2. Utilidades de los niveles de ejecución

Una de las mejores formas de configurar los niveles de ejecución es usando *initscript utility*. Estas herramientas están diseñadas para simplificar las tareas de mantener archivos en la jerarquía del directorio `SysV init` y descargan a los administradores de sistemas de tener que directamente manipular numerosos enlaces simbólicos en los subdirectorios de `/etc/rc.d/`.

Red HatLinux ofrece tres de tales utilidades:

**/sbin/chkconfig:** La utilidad `/sbin/chkconfig` es una herramienta de línea de comandos sencilla para mantener la jerarquía del directorio `/etc/rc.d/init.d`.

**/sbin/ntsysv:** La utilidad basada en ncurses `/sbin/ntsysv` provee de una interfaz interactiva basada en texto, que muchos encuentran más fácil de usar que `chkconfig`.

**Herramienta de configuración de servicios:** El programa de interfaz gráfica Herramienta de configuración de servicios `system-config-services` es una utilidad flexible basada en GTK2 para la configuración de niveles de ejecución.

### 4.3. Servicios básicos

En esta sección, se verá qué servicios están funcionando en nuestro sistema, para decidir cuáles son realmente necesarios y luego eliminar el resto.

Para saber qué servicios se debe correr, primeramente, hay que conocer qué puertos ocupa cada servicio. Los servicios más comunes, utilizan puertos que están documentados y pertenecen a un estándar (Estos puertos corresponden a el rango de números entre el 1 y el 1024), los cuales pueden ser revisados en el archivo `/etc/services`. Por ejemplo, podemos ver de allí, que el puerto que ocupa el servicio *SSH* es el 22, el de una página web (protocolo *http*) es el 80 o el de *rsync* es el 873.

#### 4.3.1. Intervención del sistema

Una utilidad que puede servir para ver qué puertos están siendo utilizados dentro de nuestro sistema, es `netstat`. Por ejemplo:

```
> netstat -vatp
Active Internet connections (servers and established)
Proto Rec Send Local Address      Foreign Address   State             PID/Program name
tcp    0  0  0.0.0.0:32768      0.0.0.0:*          LISTEN            817/rpc.statd
tcp    0  0  127.0.0.1:32769  0.0.0.0:*          LISTEN            983/xinetd
tcp    0  0  0.0.0.0:3306      0.0.0.0:*          LISTEN            1065/mysqld
tcp    0  0  0.0.0.0:111      0.0.0.0:*          LISTEN            789/portmap
tcp    0  0  0.0.0.0:6000     0.0.0.0:*          LISTEN            1532/X
tcp    0  0  0.0.0.0:80       0.0.0.0:*          LISTEN            1073/httpd
tcp    0  0  0.0.0.0:21      0.0.0.0:*          LISTEN            983/xinetd
```

```

tcp 0 0 0.0.0.0:22          0.0.0.0:*          LISTEN      949/sshd
tcp 0 0 0.0.0.0:443        0.0.0.0:*          LISTEN      1073/httpd
tcp 1 0 200.86.109.126:33693 209.73.164.146:80 CLOSE_WAIT  1887/opera
tcp 0 0 200.86.109.126:32796 200.1.19.51:22     ESTABLISHED 1639/ssh

```

Lo cual indica que esta máquina está ocupando servicios tales como xinetd, mysqld, httpd y sshd entre otros.

Tomemos en cuenta la sexta línea... la que dice

```

tcp 0 0 0.0.0.0:80         0.0.0.0:*          LISTEN      1073/httpd

```

La dirección local es la 0.0.0.0, o sea, todas las interfaces están disponibles. El puerto local es el 80, o puerto estándar para el servidor web.

El hecho de que está escuchando (LISTEN) en todos los interfaces es significativo. En este caso, sería bajo (*localhost*), eth0, y eth1. Las conexiones del servidor web se pueden acceder de cualquiera de estas interfaces. Si un usuario en este sistema tuviera una conexión PPP, entonces el demonio del servidor web estaría escuchando en esa interfaz también (la *ppp0*). La "dirección foránea" también es 0.0.0.0, lo que significa que puede venir de "dondequiera".

#### 4.3.2. Estableciendo los servicios necesarios

Centremos la atención ahora sólo en los servicios que están escuchando en este momento dentro de nuestro sistema.

```
# netstat -tap |grep LISTEN
```

```

tcp 0 0 *:32768            ::: LISTEN      817/rpc.statd
tcp 0 0 localhost.localdo:32769 ::: LISTEN      983/xinetd
tcp 0 0 *:mysql            ::: LISTEN      1065/mysqld
tcp 0 0 *:sunrpc           ::: LISTEN      789/portmap
tcp 0 0 *:x11              ::: LISTEN      1532/X
tcp 0 0 *:http             ::: LISTEN      1073/httpd
tcp 0 0 *:ftp              ::: LISTEN      983/xinetd
tcp 0 0 *:ssh              ::: LISTEN      949/sshd
tcp 0 0 *:https            ::: LISTEN      1073/httpd

```

Note que esta configuración, puede ser una de las miles posibles que puede tener en su sistema, por lo que no se asuste si no tiene las mismas líneas.

De éstas líneas, podemos destacar las que dicen que está corriendo el servicio x11 (Interfáz gráfica del sistema) y la de ssh (para aceptar conexiones a nuestra máquina de manera remota). Todos los demás servicios podemos por lo tanto deshecharlos, ya que no serían útiles.

#### 4.3.3. Apagando servicios

El paso siguiente, es encontrar dónde cada servidor de nuestra lista de matanza está encendiendo. Si no es obvio de la salida del netstat, utilicemos ps, find, grep o locate para encontrar más información del "programa conocido" o PID Info en la columna pasada.

Si el nombre del servicio o el número de acceso no parece familiar a usted, usted puede ser que consiga una breve explicación verdadera en su archivo de `/etc/services`.

El comando `chkconfig` es muy útil para controlar los servicios que es iniciado a través de los `scripts` del `init` (véase el ejemplo de más abajo). También, donde se utiliza el `xinetd`, puede controlar esos servicios también.

`chkconfig` puede decirnos qué servicios están configurados en el sistema para funcionar dependiendo del *runlevel* de ejecución, pero no necesariamente todos los servicios que de hecho están funcionando en este momento, son los que salen listados allí, ya que se pueden iniciar por otros medios, como por ejemplo, desde el `rc.local`. `chkconfig` es una herramienta de configuración, más un sistema en tiempo real que revisa qué servicios están funcionando

Hay varios lugares y maneras de iniciar servicios de sistema. Veamos las maneras más comunes para realizar esto. Los servicios de sistema son comenzados típicamente por los `scripts` del “*init*”, o por el `xinetd`.

#### 4.3.3.1. Deteniendo servicios del `init`

Los servicios de *init* son típicamente inicializados de manera automática, durante el proceso de *boot*, o durante un cambio de *runlevel*. Hay un esquema de nombramiento que utiliza `symlinks` para determinarse qué servicios deben ser comenzados, o parado, en cualquier *runlevel* dado. Las escrituras ellos mismos deben estar en `/etc/init.d/`

Usted puede conseguir un listado de estos `scripts`:

```
# ls -l /etc/init.d/ | less
```

Luego podemos detener el servicio (como `root`)

```
# /etc/init.d/<$NOMBRE_DEL_SERVICIO> stop
```

Donde `$NOMBRE_DEL_SERVICIO` es el nombre del *script* del *init*, que a menudo (pero no siempre) es igual al nombre del mismo servicio.

Esto funciona ahora solamente, para este servicio en particular, pero una vez que se reinicie el sistema, o se cambie de *runlevel*, volverá al estado por defecto. Para cambiar esto, es necesario hacer un proceso de dos pasos para los servicios del tipo `init`.

**chkconfig** se puede utilizar para considerar qué servicios se comienzan en cada *runlevel*, y apagar cualquier servicio innecesario. Para ver todos los servicios que están bajo su control, basta con escribir

```
# chkconfig --list | less
```

La primera columna es el nombre del servicio, y las columnas restantes son los distintos *runlevels*. Necesitamos generalmente solamente preocuparnos de los *runlevels* 3 (Modo Full multiusuario) y 5 (Nivel 3 + Ambiente Gráfico X11). Los servicios del `xinetd` no tendrán columnas, ese aspecto será controlado por el mismo `xinetd`.

Luego podemos apagar algunos servicios de la siguiente manera:

```
# chkconfig nfs off
# chkconfig sendmail off
```

Obviamente, Fedora core posee una herramienta que puede facilitarnos la vida para realizar estas operaciones **system-config-services**

#### 4.3.3.2. Xinetd

`xinetd` es un reemplazo del antiguo `inetd`. La configuración puede estar en el archivo `/etc/xinetd.conf`, o archivos individuales en el directorio `/etc/xinetd.d/`. La configuración de servicios individuales estará en los archivos individuales debajo de `/etc/xinetd.d/*`.

Para apagar servicios del `xinetd`, se hace suprimiendo la sección de configuración correspondiente, o el mismo archivo. Otra opción es usando un editor y simplemente fijando `disable = yes` para el servicio apropiado. O usando el `chkconfig`. Entonces, el `xinetd` necesitará ser reiniciado. Vea *man xinetd* y *man xinetd.conf* para verificar la sintaxis y las opciones de configuración. Un ejemplo de de una configuración `xinetd`, puede ser:

```
# default: on
# description: The wu-ftpd FTP server serves FTP connections. It uses \
#             normal, unencrypted usernames and passwords for authentication.
service ftp
{
    disable                = no
    socket_type            = stream
    wait                  = no
    user                  = root
    server                = /usr/sbin/in.ftpd
    server_args           = -l -a
    log_on_success        += DURATION USERID
    log_on_failure        += USERID
    nice                  = 10
}
```

Para ver de manera rápida qué servicios pueden estar habilitados, basta con utilizar:

```
# grep disable /etc/xinetd.d/* |grep no
/etc/xinetd.d/sgi_fam:  disable = no
/etc/xinetd.d/wu-ftpd:  disable = no
```

## 5. El kernel en Linux

El kernel (núcleo) del sistema es su parte medular, que interactúa directamente con la mayor parte de los dispositivos y ofrece las abstracciones familiares como archivos y procesos. Como tal, es una parte crítica del sistema, y también la que más íntimamente depende del hardware instalado en la máquina.

El paquete del núcleo en esta máquina tiene 57 MiB de archivos a instalar, lo cual sería imposible de acomodar en una máquina razonable. Y precisamente una de las ventajas de Linux es que es capaz de correr en máquinas muy limitadas.

Como una forma de resolver el problema que significaría el tener que tener un núcleo capaz de manejar directamente la enorme variedad de configuraciones que Linux soporta, se inventó la idea de manejar *módulos*, piezas del núcleo que se agregan al sistema en funcionamiento. Núcleos modernos incluso son capaces de cargar módulos según demanda, basta hacer referencia a la funcionalidad requerida (un dispositivo, un sistema de archivos, e incluso manejo de protocolos) para que los módulos requeridos se carguen automáticamente.

### 5.1. Comandos de manejo de módulos

Para manejar módulos hay varios comandos, de los cuales trataremos sólo los de más alto nivel. Hay comandos adicionales, pero son de interés sólo de desarrolladores del núcleo mismo.

**lsmod**

Muestra los módulos actualmente cargados en el núcleo. Da el nombre de cada uno, su tamaño, el número de usuarios directos, si está o no en uso (o está sujeto a ser eliminado automáticamente), y la lista de módulos que dependen de él actualmente (esto es independiente de los usuarios directos).

**modinfo** módulo...

Muestra información general sobre el módulo, como el nombre del archivo, una descripción somera, autor, licencia, y los parámetros que el módulo acepta con sus tipos.

**depmod** [-ae] [versión]

Los módulos pueden depender unos de otros, este comando construye el archivo `modules.dep` que usa `modprobe(8)` para cargarlos en el orden adecuado. La opción `-a` especifica ubicar módulos en todos los directorios mencionados en el archivo `/etc/modules.conf` (`modules.conf(5)`). La opción `-e` solicita se muestren todos los símbolos que causan errores en el proceso. De no darse versión del núcleo, procesa los módulos para el núcleo que está corriendo. Fedora core corre este comando desde `/etc/rc.sysinit`, cuando el sistema se inicia.

**modprobe** módulo...

Carga los módulos indicados (y todos los módulos que requieran, según indicado en `modules.dep`) y los inicializa.

**rmmod** [-ar] [módulo...]

Descarga los módulos indicados, si están libres. Con `-r` descarga los módulos de los que dependen también, en caso que queden libres. La opción `-a` especifica eliminar todos los módulos en desuso, aunque esto no es completamente funcional.

```
/etc/modules.conf
```

El archivo central de configuración del sistema de módulos, El uso principal es asociar módulos específicos a funcionalidades requeridas. Por ejemplo, para asociar la interfaz de red `eth0` al módulo `eeepro100` se ingresa:

```
alias eth0 eeepro100
```

Se pueden asociar parámetros a un módulo o a un alias, de forma que al cargarse el módulo tome esos argumentos. Para una tarjeta de red NE 2000 ISA podría ser:

```
alias eth1 ne
options eth1 io=0x320 irq=11
```

También se pueden especificar comandos a ejecutar antes o después de cargar o descargar un módulo dado:

```
pre-install modulo comando
post-install modulo comando
pre-remove modulo comando
pre-install modulo comando
```

## 5.2. El sistema de archivos inicial

Es posible que el funcionamiento del sistema requiera módulos que no son de uso corriente, como es el caso de sistemas con discos SCSI. En tal caso se requiere cargar los módulos requeridos incluso antes de comenzar a usar archivos en el disco. Esto se resuelve mediante el mecanismo de un sistema de archivos mínimo (`initrd`) que se carga en memoria junto con el núcleo. Inicialmente el núcleo usa este sistema de archivos inicial, cargando los módulos requeridos de allí, para luego liberarlo y montar los sistemas de archivos del disco.

```
mkinitrd imagen versión
```

Crea el archivo que contiene la imagen comprimida del sistema de archivos inicial para el núcleo de la versión dada. Fedora core coloca la imagen para el núcleo *versión* en `/boot/initrd-versión.img`.

## 6. Procesos y Señales

Un sistema Linux típico puede prestar muchos servicios simultáneamente, puede ser servidor de web, al tiempo que es servidor de correo electrónico, puede atender varios usuarios y cada usuario puede estar realizando simultáneamente diversas acciones. Por esto Linux es llamado un sistema multitarea.

A cada acción en un sistema Linux se le llama proceso. Un proceso abstrae una acción que el sistema debe realizar, independiente del momento en que debe ejecutarse. En esta sección se explica como puede controlar procesos y como puede aprovechar al máximo las capacidades multitarea de Linux, por ejemplo realizando diversas labores simultáneamente, o haciendo que la ejecución de un programa continúe después de que usted cierra su sesión (por ejemplo si se trata de un programa que debe correr durante varias horas o días), o incluso programando el inicio de procesos en momentos en los que usted no tiene una sesión abierta (por ejemplo durante la noche —claro está mientras el computador esté encendido a la hora que programe la tarea).

Abstracción de una labor que el sistema debe realizar, un comando creará al menos uno de estos, pueden recibir señales enviadas por el comando `kill(1)`.

### 6.1. Procesos

Cada proceso tiene asociado un número que lo identifica, un estado que indica como está operando, un grupo que lo asocia con otros procesos, una prioridad que determina su "importancia" con respecto a otros procesos y un dueño que puede controlarlo (normalmente el dueño es el usuario que inicia el proceso). Todos los procesos comparten el procesador —su computador normalmente tendrá un sólo procesador—, para lograrlo, cada proceso emplea el procesador durante un intervalo corto de tiempo y después duerme o se bloquea para dar posibilidad a otro proceso de emplearlo (el orden en el que se ejecutan depende de la prioridad de cada proceso). Normalmente junto con cada programa iniciado por el usuario se inicia un proceso, que a su vez puede iniciar otros procesos formando así un árbol; puede examinar tal árbol con el programa `ps(1)`. Existen también procesos que no son iniciados explícitamente por un usuario, por ejemplo procesos iniciados durante el arranque del sistema o por X-Window, tales procesos generalmente pueden ser controlados sólo por el administrador del sistema —quien también podría controlar los procesos de los usuarios.

#### 6.1.1. Ejecutando procesos en segundo plano

Cada programa o tubería que inicie desde el intérprete de comando se ejecutará en un nuevo proceso que por defecto estará en primer plano, es decir que `bash` suspenderá su ejecución y la reanudará cuando el programa que inició termine.

Si desea iniciar un programa (o una secuencia de programas unidos por tuberías) en segundo plano, agregué al final del comando un espacio y el carácter '&'. Esto es útil cuando debe ejecutar un programa no interactivo que toma bastante tiempo en completarse, porque mientras la ejecución del programa se completa puede continuar trabajando en el intérprete de comandos —el programa que inicie se ejecutará en segundo plano mientras `bash` continúa ejecutándose en primer plano.

Por ejemplo la conversión de DVI a PostScript de un documento grande puede tomar bastante tiempo, para realizar la labor en el fondo puede emplear:

```
dvi2ps -o salida.ps entrada.dvi &
```

Cuando inicia un programa (o una tubería) en segundo plano, *bash* reanuda su ejecución inmediatamente, presenta el número de tarea que asignó al comando y a continuación el número del proceso.

```
[2] 2116
```

### 6.1.2. Listando los procesos del sistema

Además de *pstree(1)*, un usuario puede ver sus procesos con el programa *ps(1)* (con la opción *-e*, *ps(1)* muestra todos los procesos del sistema). Junto con cada proceso *ps(1)* presenta:

- identificación del proceso
- la terminal en la que presenta información, en caso de que funcione de forma interactiva (por ejemplo una consola virtual como *tty1* o una terminal de X-Window como *pts/0*)
- el estado del proceso
- el tiempo que ha usado el procesador —el resto del tiempo que el proceso haya existido ha estado durmiendo o esperando algún evento o recurso.

Para examinar interactivamente los procesos de un sistema pueden emplearse los programas *top* o *gtop*, los cuales además de presentar los procesos y refrescar continuamente sus estadísticas, permiten enviar señales a cada proceso (entre otras diferencias *top* funciona en modo texto mientras que *gtop* es una aplicación Gnome).

## 6.2. Señales

En ocasiones usted deseará terminar algún proceso, por ejemplo porque deja de responder o tarda demasiado en completarse; para hacerlo puede emplear el programa *kill(1)* para enviarle una señal de terminación. Una señal es como un "llamado de atención" que se hace a un proceso en situaciones excepcionales (por ejemplo errores), pueden ser producidas por otros procesos, por el usuario o por el sistema operativo y en la mayoría de los casos conducen a la terminación del proceso que recibe la señal. Hay diversos tipos de señales, cada una tiene un número, un nombre que la identifica y una acción predefinida (que generalmente puede ser cambiada por el proceso). Un usuario puede enviar una señal a un proceso con el programa *kill(1)* seguido de la señal que enviará y del proceso que la recibirá:

```
kill -[señal] [proceso]
```

ejemplo:

```
kill -SIGTERM 945
```

Este ejemplo envía la señal `SIGTERM` al proceso con identificación 945 (en vez de `SIGTERM` pudo haberse usado 15 que es el número que corresponde a esa señal). Puede consultar un listado de todas las señales y sus números con `kill -l`.

A continuación se presenta una breve descripción de algunas señales comúnmente empleadas por usuarios:

**15 SIGTERM** Esta señal solicita la terminación del proceso que la recibe.

**9 SIGKILL** Esta señal termina el proceso que la recibe de forma inmediata. Empleela sólo para detener procesos que no terminan con la señal `SIGTERM`.

**2 SIGINT** Es la misma señal que se produce cuando un usuario en un programa interactivo presiona, Control-C para solicitar su terminación.

**3 SIGQUIT** La misma señal producida por Control- su efecto es análogo al de `SIGINT` pero además actúa como si el programa hubiera provocado algún error interno (volcando el contenido de memoria a un archivo core).

**20 SIGTSTP** La misma señal producida por Control-z, su efecto es suspender la ejecución de un proceso —para reanudarla después.

**18 SIGCONT** Reanuda un proceso suspendido previamente por la señal `SIGTSTP`.

**1 SIGHUP** Esta señal es enviada por bash a todas las tareas que se ejecutan en segundo plano, cuando el usuario cierra la sesión (por ejemplo al cerrar una terminal en X-Window o cuando sale de su sesión desde una consola virtual).

### 6.3. Creando y verificando el sistema de archivos ext3

El sistema de archivos por defecto utilizado en Linux es `ext3`, el cual es un sistema de archivos `ext2` con bitácora (o journaling). Un sistema de archivos con bitácora como es `ext3` hace que el sistema sea mucho más robusto ante cortes de luz y un apagado brusco del equipo. Cuando posteriormente arranquemos el sistema de nuevo, esté tardará menos en estar disponible porque no ejecutará el “`fsck`”.

#### 6.3.1. Migrando los sistemas de archivos de ext2 a ext3

Desmontamos la partición que queremos migrar. (Suponemos `/dev/hda3`) Para convertir la partición de `ext2` a `ext3` ejecutaremos...

```
# tune2fs -j /dev/hda3
```

Ahora debemos editar `/etc/fstab` e indicar que el nuevo sistema de archivos para la partición `/dev/hda3` es `ext3`.

El último paso es montar la unidad.

Para formatear una unidad con soporte para `ext3`, utilizaremos simplemente con el comando:

```
# mke2fs -j /dev/hdaX
```

Donde `/dev/hdaX` es la nueva partición

### 6.3.2. Reparar un sistema de archivos ext3 que está dañado

Para reparar un sistema de archivos ext3 dañado ejecutaremos...

```
# e2fsck -fy /dev/hdaX
-f Para comprobar
-y Asumir SI a todas las preguntas
```

El comando “e2fsck” pertenece al paquete e2fsprogs. Al reparar el sistema de archivos ext3 ya podemos volver a montarlo como ext2. También podemos ocupar para estas intervenciones el comando *fsck*

El sistema de archivos ext3 puede ser montado como ext2 siempre y cuando haya sido desmontado de manera correcta. Esto es una opción muy interesante, sobre todo si se tiene en cuenta que RedHat ofrece la opción de arrancar como el modo rescate (recue mode) desde CD y este no soporta *ext3*. Resultará imposible montar como *ext2* un sistema *ext3* que ha sido mal desmontado, ya que *ext2* detectará datos en la bitácora y no sabe como manejarlos.

## 6.4. Sistemas de Volúmenes Lógicos (LVM)

LVM es un módulo que se le agrega al núcleo Linux y genera una abstracción entre los discos físicos y los dispositivos para accederlos. Con la ayuda de herramientas de administración, el administrador puede acceder a los beneficios de utilizar LVM.

Básicamente lo que se logra es tener un nuevo dispositivo que apunta no a un disco o una partición, sino a un grupo de discos y particiones como un todo (manejar muchos espacios de disco como si fuera un único disco).

Entrando en detalle vamos a encontrarnos inicialmente con tres nuevos conceptos que utiliza el LVM:

**Volúmenes Físicos (VF)** Son los discos o particiones de un disco

**Volúmenes Lógicos (VL)** Son dispositivos donde se pueden crear sistemas de archivos

**Grupo Volumen (GV)** Es un área donde se juntan los VF y VL.

Bueno, veámoslo de otra manera... el *Volumen Físico* es sencillo, es el pedazo de disco que puedo tocar, ver, sentir, tirarlo contra la pared, pisarlo, etc... por otro lado, el *Grupo Volumen* es como si fuera una canasta de manzanas, siendo cada manzana un *VF*, por otro lado los *Volúmenes Lógicos* son bolsas virtuales que contienen una cantidad modificable de manzanas de la canasta.

Siguiendo con esta analogía, el Administrador va a poder sacar manzanas de la canasta (esto sería la acción de achicar el VL), o cambiar alguna manzana utilizada (que ya esta viejita y empezando a mostrar algunos hongos) por otra que no esta en uso, sacándola de la bolsa y reemplazándola por otra manzana (más nueva y sin uso). Esta nueva manzana podrá ser una manzana que ya estaba dentro de la canasta (un *VF* no utilizado) o una manzana que se acaba de meter dentro de la canasta (un nuevo *VF* que se agrego al *GV*). Esto último es el proceso de agrandar el *Volumen Lógico*. Dentro de la canasta podrán existir más de una bolsa para poner manzanas.<sup>9</sup>

Las manzanas son los *Volúmenes Físicos*, la canasta es el *Grupo Volumen* y las bolsas son los *Volúmenes Lógicos*. En los *Volúmenes Lógicos* es donde se pueden hacer los sistemas de archivos ya que, a diferencia de los *VF* o *GV*, son accesibles desde un dispositivo.

<sup>9</sup>Referencia: <http://www.xtech.com.ar/articulos/lvm/html>

Los *Volúmenes Lógicos* son los que contienen a los *Volúmenes Físicos*, y los *Grupo Volumen* son quienes contienen a los *Volúmenes Lógicos*.

#### 6.4.1. Ocupando LVM

Para poder ocupar *LVM*, debemos hacer lo siguiente:

1. Preparar los discos rígidos o particiones como *VF* (Lavar las manzanas recién traídas de la verdulería)
2. Crear un *GV* asignando uno o varios *VF* (poner las manzanas limpias dentro de la canasta)
3. Crear un *VL* asignado al *GV* (meter manzanas dentro de la bolsa y dejar la bolsa en la canasta para que no se arruine)
4. Crear un sistema de archivos (filesystem) sobre el *VL*

##### 6.4.1.1. Volúmenes Físicos

El proceso de crear un *volumen físico* es simple y rápido, como ya se mencionó, un volumen físico puede ser un disco o una partición, y para ello se ocupará el comando *pvcreate*

Por ejemplo:

```
pvcreate /dev/hdb1  
pvcreate /dev/hdc
```

Se deberá ejecutar este comando por cada uno de las particiones o discos que se quieran meter dentro de un *GV*. Hay que tener en cuenta, que este proceso es destructivo, o sea, una vez que se ejecutó *pvcreate* sobre un dispositivo, los datos que estaban contenidos allí, nunca más podrán ser recuperados.

Si el disco donde estas creando el volumen físico tiene bloques dañados es muy posible que el *pvcreate* no lo detecte, lo que hace el *pvcreate* es grabar cierta información en los primeros 512 kb. del disco/partición, pero no revisa por si hay bloques dañados. En el caso de que el *pvcreate* sí encuentre que esta dañada la primer sección del disco, dará un mensaje de error y el *VF* no se creará con éxito.

##### 6.4.1.2. Grupo Volumen

Una vez preparados todos los discos disponibles hay que asignarlos a un *Grupo Volumen*, para crear un *GV* se necesita por lo menos tener un *VF* disponible. El comando que se utiliza para ello es *vgcreate*

Por ejemplo:

```
vgcreate gv1 /dev/hdb1 /dev/hdc
```

Al crear un *grupo volumen* se genera en el directorio */dev* un nuevo directorio con el nombre asignado al *grupo volumen*, por lo tanto es importante que el nombre de este *grupo volumen* sea algo que también pueda ser el nombre de un directorio, igualmente se recomienda por el bien del administrador que sea algo corto y simple.

### 6.4.1.3. Volúmenes Lógicos

Por último ya estamos listos para asignar todo este espacio disponible a un *Volumen Lógico* y así poder crear un sistema de archivos sobre él. Para realizar esto se deberá utilizar el comando `lvcreate`

Por ejemplo:

```
lvcreate -L 40G -n v11 gv1
```

El tamaño que se le da al *Volumen Lógico* es un valor igual o menor al tamaño total disponible en el *Grupo Volumen* (o sea la sumatoria de bytes de los *VF* que están asignados a *GV*). Se pueden utilizar las letras “M” (Megabytes), “G” (Gigabytes) o “T” (Terabytes).

El resultado de este comando es un nuevo dispositivo llamado `/dev/gv1/v11` el cual puede ser utilizado como argumento del `mkfs`

Se recomienda utilizar sistemas de archivo con registro (*Journaling File Systems*), por ejemplo *XFS*, *ext3*, *ReiserFS*, *JFS*, ya que los sistemas de archivos creados sobre dispositivos de *LVM* tienden a ser grandes y una revisión de este tipo sistema de archivos puede ser extremadamente lenta y tediosa.

## 7. Automatizando tareas de administración con BASH

Descendiente del *Bourne Shell*, *bash* es un producto *GNU*, el **Bourne Again Shell**. Es el interfaz estándar de línea de comandos en la mayoría de las máquinas *Linux*. Potencia la interactividad, soportando edición en línea de comando, capacidad de completar o recordar automáticamente un comando, etc.

La shell no sólo es capaz de interpretar comandos, puede programarse usando archivos de texto que ésta interpretará, se llaman **scripts** y la shell ofrece construcciones y facilidades para facilitar su programación. Los *scripts de shell* son muy útiles para ciertos tipos de tareas:

- Tareas administrativas: algunas partes de los sistemas UNIX son scripts de shell, para poder entenderlos y modificarlos es necesario tener alguna noción sobre la programación de scripts.
- Tareas tediosas que sólo se van a ejecutar una o dos veces, no importa el rendimiento del programa resultante pero sí conviene que su programación sea rápida.
- Hacer que varios programas funcionen como un conjunto de una forma sencilla.
- Pueden ser un buen método para desarrollar prototipos de aplicaciones más complejas que posteriormente se implementarán en lenguajes más potentes.
- Conocer a fondo la shell aumenta tremendamente la rapidez y productividad a la hora de usarla, incluso fuera de los scripts.

### 7.1. Principios de programación con BASH

Para poder utilizar de la mejor manera posible nuestro entorno texto de Linux, utilizaremos la programación en *bash* y así se podrán automatizar distintas tareas. Para ello, se explicará a continuación algunos principios básicos de programación con *bash*

#### 7.1.1. Variables de entorno

Hay una serie de variables que afectan al comportamiento de la shell, tanto a la hora de trabajar de forma interactiva, como desde los scripts que ésta interpreta. Estas variables pueden ser accedidas y modificadas en la línea de comandos y también en los scripts.

Se puede ver el valor de todas las variables de entorno definidas en un momento dado invocando al comando **set** sin argumentos.

Algunas variables especialmente útiles y su significado:

**\$HOME** Directorio “*home*” del usuario.

**\$PATH** Rutas en las que la shell busca los ejecutables cuando se invoca un comando.

**\$?** Esta variable contiene el valor de salida del último comando ejecutado, es útil para saber si un comando ha finalizado con éxito o ha tenido problemas. Un ‘0’ indica que no ha habido errores, otro valor indica que sí ha habido errores.

**\$UID** Identificador del usuario que ejecuta el script.

**#!** Identificador de proceso del último comando ejecutado en segundo plano.

No es necesario declarar las variables, basta con asignar un valor a una variable para crearla. Para acceder al valor que contiene una variable se usa el caracter \$, de la siguiente forma:

```
variable = `date`      \\ OJO con las comillas
echo $variable
```

otra forma de interpretar dentro de \$variable la fecha de hoy, es mediante:

```
variable = $(date)
echo $variable
```

Como cualquier programa, los scripts pueden recibir parámetros en la línea de comandos, los parámetros recibidos se guardan en una serie de variables que el script puede consultar. Estas variables tienen los siguiente nombres:

\$1 \$2 \$3 .... \$10 \$11 ....

La variable \$0 contiene el nombre con el que se ha invocado al script.

El comando **shift** mueve todos los parámetros una posición a la izquierda, esto hace que el parámetro que haya en \$1 desaparezca, y sea reemplazado por el que había en \$2.

La variable \$# contiene el número de parámetros que ha recibido el script.

\$\* contiene todos los parámetros juntos en una sola cadena.

### 7.1.2. Uso de las comillas

En general las comillas se usan para prevenir que la *shell* intérprete ciertos caracteres dentro de una cadena y para que tome una cadena con espacios como una sola palabra.

#### 7.1.2.1. Comillas dobles

En general los caracteres especiales no son interpretados cuando están entre comillas dobles. Sin embargo algunos de ellos sí son interpretados:

\$ Está permitido referenciar variables dentro de las comillas dobles.

\ Se pueden escapar caracteres.

` Se puede realizar sustitución de comandos, esto es, ejecutar un comando y sustituir la cadena por su salida.

#### 7.1.2.2. Comillas simples

Dentro de las comillas simples todos los caracteres son interpretados literalmente, ninguno de los caracteres especiales conserva sus significado dentro de ellas.

#### 7.1.2.3. Comillas invertidas

Poner una cadena entre comillas invertidas supone ejecutar su contenido como un comando y sustituir su salida.

### 7.1.3. Tests

Un **test** es una expresión que permite evaluar si una expresión es verdadera o falsa. Los tests no sólo operan sobre los valores de las variables, también permiten conocer, por ejemplo, las propiedades de un archivo.

Los tests se usan, principalmente, en la estructura *if then else fi* para determinar qué parte del script se va a ejecutar. Un *if* puede evaluar, además de un *test*, otras expresiones, como una lista de comandos (usando su valor de retorno), una variable o una expresión aritmética. Este es un ejemplo del uso de *if*:

```
if grep snoopy archivo-lindo.txt > /dev/null
then
    echo "archivo-lindo.txt contiene la palabra snoopy"
else
    echo "archivo-lindo.txt no contiene la palabra snoopy"
fi
```

Hay dos formas distintas de escribir un test, `[ ]` y `[[ ]]`. No son equivalentes, (por ejemplo los operadores `&&` — *iy sólo funcionan en la última*), pero de momento las diferencias son irrelevantes.

```
if [[ test ]]
then
    comando
else
    comando
fi
```

### 7.1.4. Estructuras de control

Como en cualquier lenguaje de programación, la shell ofrece estructuras que permiten controlar el flujo de ejecución de los scripts.

#### 7.1.4.1. Bucle for

Su sintaxis básica es la que sigue:

```
for var in lista de valores
do
    comandos
done
```

La variable `$var` toma el valor del siguiente valor de la lista en cada iteración. Un ejemplo:

```
for i in $(ls *.sh)
do
    if [ -x "$i" ]
    then
        echo "El archivo \"$i\" es ejecutable"
    fi
done
```

#### 7.1.4.2. Bucle while

Su sintaxis es la siguiente:

```
while [ condicion ]
do
    comandos
done
```

La condicion puede ser, al igual que en un if, cualquier test, comando o expresión, el bucle se ejecutará mientras que la condición devuelva verdadero, es decir, cero.

En los bucles break y continue tienen el mismo funcionamiento que en otros lenguajes. break termina el bucle y continue salta a la siguiente iteración.

#### 7.1.4.3. Case

Como en otros lenguajes case sirve para ejecutar una zona de código u otra, en función del valor de una expresión o variable:

```
case $var in
    valor ) comandos ;;
    valor2 ) comandos ;;
esac
```

El funcionamiento de case puede verse en los scripts de inicio del sistema, lo usan para discernir si han sido llamados con los parámetros start restart stop o algún otro.

#### 7.1.5. Globbing

El globbing, también conocido por “filename expansion”, es decir, expansión de nombres de archivos, es el tratamiento que hace la *shell* cuando encuentra un nombre de archivo. Cuando se le indica a la *shell* el nombre de un archivo, algunos caracteres tienen un significado especial que la *shell* interpreta antes de hacer lo que tenga que hacer con ese nombre. Los caracteres son estos:

- \* Corresponde con cualquier secuencia de cero o más caracteres, con la excepción de los archivos cuyo nombre empieza con un punto.
- ? Corresponde con cualquier caracter, una sola vez.
- [ ] Corresponde con cualquiera de los caracteres o rangos de caracteres que contenga.
- ^ Niega la expresión que le sigue.

Contiene varias expresiones separadas por comas y corresponde a cualquiera de ellas.

## 7.2. Creación de Scripts

Ahora que se tiene un poco más claro cuál es el concepto de programación en *bash*, se pueden empezar a crear rutinas que automaticen algunas tareas que hagamos constantemente.

La primera línea de los scripts debe ser:

```
#!/bin/bash
o
#!/bin/sh
```

Si el archivo tiene permisos de ejecución, la primera línea indica qué intérprete se requiere para su ejecución (en este caso, *bash*), de otra manera, se deberá llamar al intérprete de comandos al ejecutar el script. Ejemplo

```
# /bin/bash /mi/lindo/script.sh
```

Todo lo que contenga el archivo script, será interpretado línea a línea de manera ordenada y secuencial por el intérprete. Las líneas que contengan el signo “#”, a partir desde ese punto, se considerarán comentarios y por lo tanto no se ejecutará nada de lo que esté escrito a la derecha de éste.

### 7.2.1. Hola mundo!

Partamos con el ejemplo más básico de todos... el script que nos muestre un “Hola mundo!”.

Con el editor de preferencia que ocupe, escribir un archivo llamado *hola.sh* que contenga lo siguiente:

```
#!/bin/bash
#Evitando la maldición...
TEXTO="Hola mundo!"
echo $TEXTO
```

Luego le damos permiso de ejecución a este archivo

```
# chmod 755 hola.sh
```

y lo ejecutamos

```
# ./hola.sh
Hola mundo!
```

Fácil, no?

Las posibilidades son inmensas y con las herramientas que ya se tienen, se puede hacer lo que uno quiera. Sólo es necesario un poco de práctica y algo de imaginación.

### 7.2.2. Un ejemplo más complejo

Con el siguiente ejemplo (respaldo automático de una Base de Datos MySQL), se pretende demostrar lo fácil que puede ser para uno automatizar tareas tediosas que puedan ser repetidas diariamente. Además se aprovecha de mostrar la utilización de crons.

```
#!/bin/bash
#MySQL Backup
# echo "0 9 * * * backup.sh" > crontab -e
#Declaración de Variables
EMAIL = "mave007@NOSPAMinf.utfsm.cl"
HOST = "localhost"
DB = "testin"
USER = "mave007"
PASSWD= "LaClave"
DIR = "/opt/Backup"
FECHA = "$(date +%Y.%m.%d)"
CONT = 10
FILE = "$DIR/testin-bak-$FECHA.sql"
MAQ = $(uname -n)
GZIP = "1" #0 no, 1 yes

#Viendo si existe $DIR
if [ ! -d $DIR ] ; then
    mkdir -p $DIR &>/dev/null
    chmod 700 $DIR &>/dev/null
fi

#Haciendo el Dump
mysqldump -h$HOST -u$USER -p$PASSWD --opt
    $DB > $FILE 2> $DIR/error

#Comprimimos??
if [ "$GZIP" = "1" ] ; then
    gzip -9 $FILE &>/dev/null
fi

#Cuántos backups hay??
num=$(ls $DIR/*.sq* |wc -l)&>/dev/null

#Dejando solo los $CONT archivos
cd $DIR
while ( [ $num -gt $CONT ] ); do
    rm -f $(ls -t |tail -n 1)
    num=$(ls $DIR/*.sq* |wc -l)&>/dev/null
done

#Enviando un email si hay error a $DIR
tam_error= $(du $DIR/error|cut -d '/' -f 1)

if ( [ $tam_error -ne 0 ] ); then
```

```
echo -e "\nError (auto)" >> $DIR/error
cat $DIR/error|mail -s"Backup :(" $EMAIL
fi
```

Luego, podemos agregar este script (*backup.sh*), para que se ejecute todos los días a las 9:00 de la mañana:

```
echo "0 9 * * * backup.sh" > crontab -e
```

## 8. El sistema de paquetes Red Hat Package Manager (RPM)

La distribución Red Hat introdujo un sistema de administración de paquetes que hoy día es usado casi universalmente en el mundo Linux. La idea es guardar en una base de datos información sobre los archivos instalados en el sistema (a qué paquete pertenecen, tamaños y permisos, dependencias entre paquetes, entre otros). El comando `rpm(1)` tiene varios modos de operación, acá nos centraremos en la instalación, desinstalación, y algunas consultas. El nombre de un paquete tiene la forma siguiente:

```
nombre-version-release.arch.rpm
```

El nombre es el nombre del paquete, la *versión* es la versión base usada para construir el paquete, *release* se refiere a la versión modificada para distribución. La arquitectura *arch* identifica la máquina para la cual se creó el paquete, o *noarch* para paquetes que no dependen de la arquitectura (como documentación). Si aparece *src* acá, se trata de un paquete con fuentes para crear el paquete binario. Los paquetes fuente contienen los fuentes originales del paquete, además de todas las modificaciones locales que aplicó la distribución, y archivos de configuración que controlan la construcción del paquete.

Se pueden encontrar paquetes adicionales a la distribución misma por ejemplo en <http://www.rpmfind.net>. En tal caso, debe tenerse cuidado de sólo instalar paquetes creados para la distribución exacta que se está usando. Distintas distribuciones tienen diferencias sutiles, con lo que paquetes extranjeros pueden producir problemas severos.

### 8.1. Instalando y Desinstalando Paquetes

#### 8.1.1. Consultas

```
rpm -q opciones de consulta
```

Las opciones de consulta incluyen `-f` (a qué paquete pertenece un archivo dado), `-l` (liste los archivos que pertenecen a un paquete dado), `-i` (dé información sobre el paquete), mientras `-v` solicita información adicional. El paquete puede darse como nombre de un paquete instalado, o con `-p` se da el archivo que lo contiene. Las opciones pueden combinarse, así `rpm -qfi /bin/bash` solicita información sobre el paquete que contiene el shell. Otras opciones importantes con `--whatprovides` (que paquete provee una funcionalidad particular, como un archivo dado) y `--whatrequires` (que paquete requiere alguna funcionalidad particular o depende de un paquete específico). Si está instalado el paquete con la base de datos completa de los paquetes de Red Hat (`rpmdb-redhat`), pueden usarse `--redhatprovides` y `--redhatrequires` para consultar la colección de paquetes de la distribución, no lo que está instalado.

### 8.1.2. Verificación

```
rpm -V opciones de verificación
```

Verifica lo que hay instalado contra la base de datos. Reporta archivos faltantes o que se hayan modificado en el paquete. Con `-Va` verifica todo (esto puede tomar largo tiempo).

### 8.1.3. Instalar y Actualizar

```
rpm -[iUF] opciones de instalación
```

Se pueden instalar (`-i`) nuevos paquetes, o actualizarlos (`-U`). La diferencia está en que al instalar queda la versión anterior, al actualizar ésta se elimina. La opción `--force` fuerza la instalación o actualización (para instalar una versión más antigua que la actual, o forzar reinstalar sobre la misma versión). Para simplificar la actualización está la opción de refrescar (`-F`), que actualiza sólo aquellos paquetes que ya están instalados. Opciones generales con `-v` (muestra las versiones de lo que se instala) y `-h` (muestra el avance del proceso). Los paquetes se pueden especificar como archivos locales, o como URLs a través de FTP o HTTP.

#### 8.1.3.1. Dependencias no resueltas

Los paquetes RPM pueden "depender" de otros paquetes, lo cual significa que requieren de la instalación de otros paquetes para poder ejecutarse adecuadamente. Si intenta instalar un paquete que tiene una dependencia no resuelta, verá lo siguiente:

```
Preparing... ##### [100%]
error: Failed dependencies:
    bar.so.2 is needed by foo-1.0-1
Suggested resolutions:
    bar-2.0.20-3.i386.rpm
```

Si está instalando un paquete oficial de Fedora core, se le sugerirá resolver la dependencia de este paquete. Encuentre este paquete en los CD-ROMs de Fedora core Linux o en el sitio FTP (o mirror) y añádalo al comando:

```
rpm -ivh foo-1.0-1.i386.rpm bar-2.0.20-3.i386.rpm
```

Si se realiza la instalación correctamente, verá lo siguiente:

```
Preparing... ##### [100%]
 1:foo          ##### [ 50%]
 2:bar          ##### [100%]
```

Si no se le sugiere resolver la dependencia, puede intentar usar la opción `--redhatprovides` para determinar el paquete que contenga el archivo requerido. Necesita instalar el paquete `rpmdb-redhat` para usar esta opción.

```
rpm -q --redhatprovides bar.so.2
```

Si el paquete que contiene el archivo `bar.so.2` se encuentra en la base de datos instalada del paquete `rpmdb-redhat` aparecerá el nombre del paquete:

```
bar-2.0.20-3.i386.rpm
```

Si desea forzar la instalación de todas maneras (no es una buena idea ya que el paquete no funcionará correctamente), use la opción `--nodeps`.

Otra forma de resolver estos problemas de dependencia es utilizar herramientas de adquisición de alto nivel discutidas mas adelante.

#### 8.1.4. Desinstalar

```
rpm -e paquetes a desinstalar
```

Se indican los paquetes a desinstalar, ya sea con el nombre del paquete únicamente o la versión completa en caso que hayan varias versiones instaladas.

##### 8.1.4.1. Dependencias no resueltas

Podría encontrarse con un error de dependencia cuando esté desinstalando un paquete si otro paquete instalado depende del que está tratando de eliminar. Por ejemplo:

```
Preparing...                               ##### [100%]
error: removing these packages would break dependencies:
       foo is needed by bar-2.0.20-3.i386.rpm
```

Para hacer que RPM ignore este error y desinstale el paquete de todos modos (que tampoco es buena idea ya que al hacerlo, el paquete que depende de él probablemente dejará de funcionar correctamente), use la opción `--nodeps`.

## 8.2. Herramientas de adquisición de alto nivel

### 8.2.1. APT - Advance Package Tool

Es un conjunto de herramientas que se utilizan para administrar paquetes de forma automatizada, de manera tal, que cuando el usuario solicita la instalación de un paquete (aplicación), el sistema también instala (o actualiza) todos los paquetes necesarios para el funcionamiento de esa aplicación (resolviendo dependencias).

Cabe señalar, que *apt-get* es un programa (paquete) que fue creado para la distribución *Debian* y debido a sus múltiples opciones y facilidad de uso se portó a Red Hat, bajo el nombre de *apt-rpm*, pero por razones históricas, se mantuvo el nombre de la aplicación en *apt-get*.

Debido a que fue creado para *Debian*, el manejo de paquetes lo hacía con el administrador de paquetes DEB. Ahora que ha sido portado a Fedora core, trabaja sobre paquete RPM... Esto quiere decir, que *apt-get* es sólo una API o cáscara que trabaja sobre los paquetes RPM y no un manejador de paquetes distinto de él.

### 8.2.1.1. sources.list

Este archivo, ubicado en el directorio `/etc/apt` contiene la información de los servidores desde donde se traerán los paquetes.

Ejemplo: Archivo `sources.list`.

Todas las líneas que comiencen con `#` son comentarios.

```
# List of available apt repositories available from ayo.freshrpms.net.
# This file should contain an uncommented default suitable for your system.
#
# See http://ayo.freshrpms.net/ for a list of other repositories and mirrors.

# Fedora Linux 3
rpm http://ayo.freshrpms.net fedora/linux/3/i386 core updates freshrpms
rpm http://ayo.freshrpms.net fedora/linux/3/i386 tupdates
rpm-src http://ayo.freshrpms.net fedora/linux/3/i386 core updates freshrpms
rpm-src http://ayo.freshrpms.net fedora/linux/3/i386 tupdates
```

### 8.2.1.2. Comandos de apt-get

Los comandos de `apt-get` siguen la siguiente estructura:

```
apt-get [opciones] comando
apt-get [opciones] install paquete [paquete... ]
```

La línea de comando puede ser una variación de los siguientes tipos básicos:

**apt-get update** Con este comando se actualizará la lista de paquetes que se encuentran en el servidor y serán bajados al computador local.

**apt-get check** Herramienta de diagnóstico, updatea el caché verificando integridad del sistema. Es recomendable ejecutarlo antes de empezar una actualización de la distribución

**apt-get install algun\_paquete** Instala algún paquete nuevo, resolviendo dependencias automáticamente. Si el paquete *algun\_paquete* ya está instalado, intentará actualizarlo.

**apt-get upgrade** Busca paquetes que estén desactualizados en el sistema y los actualiza automáticamente. Para actualizar el paquete y sus dependencias se debe utilizar el comando:

```
apt-get install paquete\_a\_actualizar
```

**apt-get dist-upgrade** Instala todos los paquetes básicos e intenta actualizar todo, instalando nuevos paquetes si es necesario. Esta es una manera más fácil de hacer una actualización de la distribución

**apt-get remove algun\_paquete** Elimina el paquete *algun\_paquete* y todos los demás paquetes que dependen de él.

**apt-get clean** Elimina los archivos que se encuentran en `/var/cache/apt` y que han sido bajados del servidor

### 8.2.2. YUM - YellowDog Updater Modified

Anteriormente las distribuciones de Red Hat usaban *RedHat Network* para liberar sus actualizaciones, este servicio tiene un precio de uso, si quieres tener tu sistema actualizado y no tienes las posibilidades para comprar el servicio de *RedHat Network*, sigue leyendo.

YUM (Yellow dog Updater, Modified) es una potente herramienta con la cual se puede instalar paquetes RPM sin preocuparse tanto, ya que calcula las dependencias faltantes y si estan dentro de alguno de los repositorios en el archivo de configuración también las instala. Puede eliminar paquetes RPM sin preocuparse por dejar tu sistema inestable, ya que una de sus políticas es precisamente eso, no dejar tu sistema inestable por eliminar un paquete. Sirve para actualizar desde un paquete hasta el sistema completo, es rápido de usar y entender.

#### 8.2.2.1. yum.conf

De manera similar a *apt*, *yum* mantiene un archivo de configuración que contiene las opciones por defecto que utilizará al momento de ejecutar.

Ejemplo: Archivo *yum.conf*.

Todas las líneas que comiencen con *#* son comentarios.

```
[main]
cachedir=/var/cache/yum
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
distroverpkg=redhat-release
tolerant=1
exactarch=1
retries=20
obsoletes=1
gpgcheck=0
assumeyes=1

# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
```

#### 8.2.2.2. El directorio *yum.repos.d*

Tal como aparece en la última línea de ejemplo del archivo *yum.conf*, se puede notar que los repositorios deberían quedar en el directorio */etc/yum.repos.d*, pese a que se pueden dejar en el mismo *yum.conf*. Se sugiere de todas maneras, que cada repositorio quede en un archivo separado en el directorio aludido, con un nombre similar a *NOMBRE.repo*... Por ejemplo *utfsm.repo* o *utfsm-updates.repo*

La sintaxis de estos archivos tienen básicamente 4 parámetros, el nombre del repositorio, la dirección de donde se bajan los paquetes y un campo que establece si este repositorio está habilitado o no para ser tomado en cuenta por *yum*

Ejemplo de *utfsm.repo* (El cual contiene los paquetes BASE de la distribución para Fedora core 3):

```
[UTFSM-base]
```

```
name=UTFSM-base
baseurl=ftp://ftp.inf.utfsm.cl/fc3
enabled=1
```

Ejemplo de `utfsm-updates.repo` (El cual contiene los paquetes con los parches de la distribución para Fedora core 3):

```
[UTFSM-updates]
name=UTFSM-updates
baseurl=ftp://ftp.inf.utfsm.cl/updates/3
enabled=1
```

### 8.2.2.3. Comandos de yum

Los comandos de yum, siguen la siguiente estructura:

```
yum [opciones] [comando] [paquete ...]
```

**install** Es usado para instalar la última versión de un paquete o grupo de paquetes, asegurando que todas las dependencias serán satisfechas. Si ningún paquete “calza” con el nombre dado, se asume que intentará calzar un “glob” de shell y será instalado.

**update** Si se utiliza sin ningún otro parametro, update actualizará todos los paquetes que están instalados. Si uno o más paquetes son especificados, *yum* sólo actualizará dichos paquetes.

**check-update** Chequea si hay nuevas actualizaciones, con un código de retorno 100 en caso de tener éxito.

**remove** También puede ser **erase** Es utilizado para remover los paquetes especificados del sistema y todos aquellos que dependen de él.

**search** Utilizado para buscar cualquier paquete que “calce” con el string dado en la descripción, sumario, empaquetador y/o nombre del paquete RPM. Muy útil para encontrar paquetes que no se conocen sus nombres, pero sí alguna palabra relacionada a él.

**clean** Usado para limpiar varias cosas que se van acumulando en el *cache* de *yum*. Algunas de las opciones que pueden dársele, son:

- headers
- packages
- cache
- metadata
- all

### 8.2.3. Construyendo Repositorios

#### 8.2.3.1. yum-arch

Para crear un repositorio *yum*, se debe crear un directorio `headers`, el cual contendrá cabeceras `.hdr` por cada uno de los paquetes y un archivo `header.info`, que tiene información general sobre todo el repositorio.

Para crear un repositorio yum de estas características, tan sólo basta con escribir:

```
yum-arch -v -s /DIRECTORIO/CON/RPMs
```

### 8.2.3.2. createrepo

Desde la versión 3 de Fedora core, se introduce un nuevo standard ante la posibilidad de crear repositorios que puedan anteder a clientes apt, como yum; los cuales a través de información llamada *METADATA*<sup>10</sup>, instruye a los clientes qué paquetes están en el repositorio.

Para utilizar esta herramienta, se puede utilizar con

```
createrepo -p -v /DIRECTORIO/CON/RPMs
```

---

<sup>10</sup>Más info en <http://linux.duke.edu/metadata/>

## 8.3. Construyendo un RPM simple

### 8.3.1. Construyendo a Partir de un SRC-RPM

Un Source RPM es un RPM que viene preparado y listo para ser compilado por nosotros y generar un RPM que podamos instalar. La herramienta a que debemos usar es `rpmbuild(8)`, por ejemplo:

```
rpmbuild --rebuild foo-2.4.5-12.src.rpm
```

`rpmbuild` generara el paquete `foo-2.4.5-12.rpm` y lo dejara ubicado en el directorio `/usr/src/redhat/RPMS{athlon,i386,i486,i586,i686,noarch}` dependiendo de la arquitectura para la cual se compilo.

### 8.3.2. Construyendo a Partir de un tar.gz

Antes que todo, para poder construir un RPM debe existir la siguiente estructura de directorios:

```
# ls -lF /usr/src/redhat
total 5
drwxr-xr-x 3 root root 1024 Aug 5 13:12 BUILD/
drwxr-xr-x 3 root root 1024 Jul 17 17:51 RPMS/
drwxr-xr-x 4 root root 1024 Aug 4 22:31 SOURCES/
drwxr-xr-x 2 root root 1024 Aug 5 13:12 SPECS/
drwxr-xr-x 2 root root 1024 Aug 4 22:28 SRPMS/
```

Ahora se necesita el código fuente del programa para el cual deseamos crear un RPM. En este caso usaremos los códigos fuentes `foo_bar.tar.gz` y lo ubicaremos en el directorio `SOURCES`.

#### 8.3.2.1. SPEC file

El SPEC file es el archivo que contiene las instrucciones de como crear un rpm y los elementos que debe incluir en el. Un SPEC file esta dividido en:

**Preambulo** En esta sección se incluye toda la información concerniente al paquete. Un ejemplo es el siguiente:

```
#
# Ejemplo de spec file para foo_bar
#
Summary: Una aplicación de sonido
Name: foo_bar
Version: 1.0
Release: 1
Copyright: GPL
Group: Applications/Sound
Source: ftp://lugar.en.la.web/foo_bar.tar.gz
URL: http://otro.lugar.en.la.web/foo_bar.html
Packager: Alumnos Linux Avanzado <alumnos@inf.utfsm.cl>
```

```
%description
Aquí ponemos una buena descripción de lo que
nuestro software hace.
```

**%prep** Mientras el preámbulo en su mayor parte es información para consumo humano, en esta sección se prepara el código para generar el rpm.

```
%prep
rm -fr $RPM_BUILD_DIR/foo_bar
tar zxvf $RPM_BUILD_DIR/foobar
```

**%build** En esta sección le indicamos como se debe construir el rpm.

```
%build
make
```

**%install** En esta sección se indica como instalar el rpm.

```
%install
make install
```

**%files** Esta sección contiene un listado de los archivos que serán incluidos en el paquete. Es importante recordar que si no esta en este listado, NO se incluirá. La subsección %doc marca los archivos ahí puestos como documentación.

```
%files
%doc README
/usr/local/bin/foo
/usr/local/bin/bar
/usr/local/man/man1/foo.1
```

Este archivo SPEC file debe ser guardado en `/usr/src/redhat/SPECS`.

### 8.3.2.2. Creando el RPM

Ahora se puede crear el RPM usando `rpmbuild(8)`. Primero nos debemos ubicar en el directorio donde se encuentran los spec files, y luego ejecutar `rpmbuild`.

```
rpmbuild -ba foo_bar.spec
```

Si todo va bien en la consola se indicara con un `exit 0`, y el RPM quedara ubicado en `RPMS` bajo la arquitectura correcta.

MAVE-PeNnY/ΛT<sub>ε</sub>X 2<sub>ε</sub>