Trabajando con sistemas de archivos

La jerarquía del sistema de archivos

Slackware Linux almacena todos sus archivos y directorios en un unico directorio `/`, generalmente denominado *“raiz”*. Esto claramente contrasta con lo que puede estar familiarizado en Microsoft Windows. Diferentes particiones de disco duro, cdroms, unidades flash usb e incluso disquetes pueden montarse dentro de directorios `/`, pero no tienen nada como *“letras de unidad”*. El contenido de estos dispositivos se puede encontrar en casi cualquier lugar, pero hay algunos valores predeterminados que Slackware configura para usted. Por ejemplo, las unidades cd-rw se encuentran con más frecuencia en `/ mnt/cd-rw`. Aquí hay algunos directorios comunes presentes en casi todas las instalaciones de Slackware Linux, y lo que puede esperar encontrar allí.

Tabla 11.1. Diseño del sistema de archivos

/	El directorio raíz, bajo el cual existen todos los demás.
/ bin	Conjunto mínimo de programas binarios para todos los usuarios.
/ boot	El kernel, initrd y otros requisitos para arrancar Slackware.
/ etc	Archivos de configuración del sistema.
/ dev	Colección de archivos especiales que permiten el acceso directo al hardware.
/ home	Directorios de usuarios donde se almacenan los archivos personales y las configuraciones.
/ media	Directorio para características de auto-montaje en DBUS / HAL.
/ mnt	Lugares para montar temporalmente medios extraíbles.
/ opt	Directorio donde se puede instalar software (típicamente propietario).
/ proc	Sistema de archivos exportado por el Kernel con información de procesos.
/ root	Directorio de inicio del usuario root.
/ sbin	Conjunto mínimo de binarios de sistema o superusuario.
/ srv	Datos específicos del sitio, como las páginas web servidas por este sistema.
/ sys	Detalles especiales de implementación del kernel.
/ tmp	Directorio reservado para archivos temporales para todos los usuarios.
/ usr	Todos los programas no esenciales, bibliotecas y archivos compartidos.
/ var	Datos con alta rotación, como por ejemplo logs.

Tipos de sistemas de archivos locales

El kernel Linux es compatible con una amplia variedad de sistemas de archivos, lo que le permite elegir entre una larga lista de características para adaptarlo a sus necesidades particulares. Afortunadamente, la mayoría de los tipos de sistemas de archivos predeterminados son adecuados para cualquier necesidad que pueda tener. Algunos sistemas de archivos están orientados a medios particulares. Por ejemplo, el sistema de archivos iso9660 se usa casi exclusivamente para medios de CD y DVD.

ext2

ext2 es el sistema de archivos más antiguo incluido en Slackware Linux para almacenar datos en discos rigidos. En comparación con otros sistemas de archivos, ext2 es simple. Es más rápido que la mayoría de los demás para leer y escribir datos, pero no incluye ninguna capacidad de journaling. Esto significa que después de una falla, el sistema de archivos debe revisarse exhaustivamente para descubrir y (con suerte) reparar cualquier error.

ext3

ext3 es el primo más joven de ext2. Fue diseñado para reemplazar ext2 en la mayoría de los escenarios y comparte la misma base de código, pero agrega soporte de journaling. De hecho, ext3 y ext2 son tan parecidos que es posible convertir uno a otro sobre la marcha sin perdida de datos. ext3 goza de mucha popularidad por estas razones. Hay muchas herramientas disponibles para recuperar datos de este sistema de archivos en caso de un error catastrófico del hardware. ext3 es un buen sistema de archivos de propósito general con soporte de journaling, pero no funciona tan bien como otros sistemas de archivos de journaling en casos específicos. Un inconveniente de ext3 es que el sistema de archivos todavía debe pasar por esta comprobación exhaustiva cada cierto tiempo. Esto se hace cuando se monta el sistema de archivos, generalmente cuando La computadora arranca, y causa un retraso molesto.

ext4

ext4 es lo último en la serie de sistemas de archivos ext. Fue diseñado para construir sobre ext3 nuevas ideas sobre lo que deberían hacer los sistemas de archivos. Si bien Slackware es compatible con ext4, debe recordar que este sistema de archivos es todavía muy nuevo (particularmente en términos de sistema de archivos) y aun se encuentra en desarrollo. Si necesita estabilidad sobre rendimiento, es posible que desee utilizar un sistema de archivos diferente, como ext3. Dicho esto, ext4 cuenta con algunas mejoras importantes sobre ext3 en el campo del rendimiento, pero muchas personas aún no confían en él para un uso estable.

reiserfs

reiserfs es uno de los sistemas con journaling más antiguos para el kernel Linux y ha sido soportado por Slackware durante muchos años. Es un sistema de archivos muy rápido especialmente adecuado para almacenar, recuperar y escribir muchos archivos pequeños. Desafortunadamente, hay pocas herramientas para recuperar datos si experimenta una falla en la unidad, y las particiones reiserfs experimentan daños con mayor frecuencia que ext3.

XFS

XFS fue aportado al kernel Linux por SGI y es uno de los mejores sistemas de archivos para trabajar con grandes volúmenes y grandes archivos. XFS usa más RAM que otros sistemas de archivos, pero si necesita trabajar con archivos grandes, su rendimiento justifica el uso de la memoria. XFS no es particularmente inadecuado para el uso de computadoras de escritorio o portátiles, pero realmente brilla en un servidor que maneja archivos de tamaño mediano a grande durante todo el día. Al igual que ext3, XFS es un sistema de archivos con journal.

JFS

JFS fue aportado al kernel Linux por IBM y es bien conocido por su capacidad de respuesta incluso en condiciones extremas. Puede abarcar volúmenes colosales, lo que lo hace especialmente adecuado para dispositivos de almacenamiento conectado a la red (NAS). La larga historia y las exhaustivas pruebas de JFS lo convierten en uno de los sistemas de archivos de journaling más confiables disponibles para Linux.

iso9660

iso9660 es un sistema de archivos diseñado específicamente para medios ópticos, como CD y DVD. Dado que los discos ópticos son medios de solo lectura, el kernel Linux ni siquiera incluye soporte de escritura para este sistema de archivos. Para crear un sistema de archivos iso9660, debe usar herramientas de usuario como ***mkisofs*** (8) o ***growisofs*** (8).

vfat

A veces es posible que necesite compartir datos entre computadoras con sistemas Windows y Linux, pero no puede transferir los archivos a través de una red. En su lugar, necesita una partición de disco compartida o una unidad flash USB. El humilde sistema de archivos vfat es la mejor opción aquí, ya que es compatible con la mayor variedad de sistemas operativos. Desafortunadamente, al ser un sistema de archivos diseñado por Microsoft, no almacena los permisos de la misma manera que los sistemas de archivos tradicionales de Linux. Esto significa que deben usarse opciones especiales para permitir que múltiples usuarios accedan a los datos en este sistema de archivos.

swap

A diferencia de otros sistemas de archivos que contienen archivos y directorios, las particiones de intercambio contienen la memoria virtual. Esto es muy útil ya que evita que el sistema se bloquee si se consume toda la RAM. En su lugar, el kernel copia partes de la RAM en swap y las libera para que otras aplicaciones las utilicen. Piense en ello como agregar memoria virtual a su computadora, memoria virtual muy lenta. Por lo general, el intercambio es a prueba de fallos y no se debe confiar para su uso continuo. Agregue más RAM a su sistema si se encuentra usando muchas paginas de intercambios.

Utilizando mount

Ahora que hemos aprendido (algunos de) los diferentes sistemas de archivos disponibles en Linux, es hora de que veamos cómo usarlos. Para leer o escribir datos en un sistema de archivos, primero se debe montar ese sistema de archivos. Para hacer esto, nosotros (naturalmente) usamos mount (8). Lo primero que debemos hacer es decidir dónde queremos ubicar el sistema de archivos. Recuerde que no hay tales cosas como letras de unidad que denotan sistemas de archivos en Linux. En su lugar, todos los sistemas de archivos están montados en directorios. El sistema de archivos base en el que instala Slackware siempre se encuentra en `/` y los demás siempre se encuentran en los

subdirectorios de /. /mnt/hd es un lugar común para localizar temporalmente una partición, por lo que usaremos eso en nuestro primer ejemplo. Para montar el contenido de un sistema de archivos, debemos decirle qué tipo de sistema de archivos tenemos, dónde lo montamos y qué opciones especiales usar.

```
darkstar:~# mount -t ext3 /dev/hda3 /mnt/hd -o ro
```

Vamos a analizar esto. Tenemos un sistema de archivos ext3 ubicado en la tercera partición del primer dispositivo IDE, y hemos decidido montar su contenido en el directorio /mnt/hd. Además, lo hemos montado de solo lectura para que no se puedan realizar cambios en su contenidos. El argumento [-t ext3] le indica a mount qué tipo de sistema de archivos estamos usando, en este caso es ext3. Esto le permite al núcleo saber qué controlador usar. A menudo, el montaje puede determinar esto por sí mismo, pero nunca está de más declararlo explícitamente. En segundo lugar, le decimos a mount dónde ubicar el contenido del sistema de archivos. Aquí hemos elegido /mnt/hd. Finalmente, debemos decidir qué opciones usar si las hay. Estos se declaran con el argumento [-o]. A continuación una lista de las opciones más comunes.

Tabla 11.2. Opciones más comunes para el comando mount

ro	read-only
rw	read-write (default)
uid	user id del dueño del contenido en el sistema de archivos
gid	group id del dueño del contenido en el sistema de archivos
noexec	evita la ejecución de archivos/binarios
defaults	valores predeterminados para el sistema de archivos

Si esta es tu primera instalación de Linux, las únicas opciones que por lo general debe preocuparle son *ro* y *rw*. La excepción a esta regla viene cuando estás tratando con sistemas de archivos que no manejan los permisos tradicionales de Linux como vfat o NTFS. En esos casos, deberá usar las opciones *uid* o *gid* para permitir que los usuarios no root accedan a estos sistemas de archivos.

```
darkstar:~# mount -t vfat /dev/hda4 /mnt/hd -o uid=alan
```

Pero Alan, ¡eso es espantoso! No quiero tener que decirle al montaje qué sistema de archivos u opciones usar cada vez que cargue un CD. Debería ser más fácil que eso. Bueno, afortunadamente, lo es. El archivo /etc/fstab contiene toda esta información para los sistemas de archivos que el instalador configura para usted y también puede agregarle algo. fstab (5) parece una tabla simple que contiene el dispositivo para montar junto con su tipo de sistema de archivos y argumentos opcionales. Vamos a ver.

```
darkstar:~# cat /etc/fstab
/dev/hda1       /               reiserfs    defaults              1   1
/dev/hda2       /home           reiserfs    defaults              1   2
/dev/hda3       swap            swap        defaults              0   0
/dev/cdrom      /mnt/cdrom      auto        noauto,owner,ro,users 0   0
/dev/fd0        /mnt/floppy     auto        noauto,owner          0   0
devpts          /dev/pts        devpts      gid=5,mode=620        0   0
proc            /proc           proc        defaults              0   0
```

Si tiene una entrada en fstab para su sistema de archivos, sólo necesita indicar el nodo del

dispositivo o la ubicación del montaje.

```
darkstar:~# mount /dev/cdrom
darkstar:~# mount /home
```

Un uso sencillo de ***mount*** es para decirle qué sistemas de archivos están montados actualmente y con qué opciones. Simplemente correr ***mount*** sin ningún argumento mostrara eso.

Sistemas de archivos de red

Además de los sistemas de archivos locales, Slackware admite varios sistemas de archivos de red como cliente y servidor. Esto le permite compartir datos entre varias computadoras de manera transparente. Vamos a discutir los dos más comunes: NFS y SMB.

NFS

NFS es el sistema de archivos de red para Linux, así como de varios otros sistemas operativos comunes. Tiene un rendimiento modesto pero admite la gama completa de permisos para Slackware. Para utilizar NFS como cliente o como servidor, debe ejecutar el demonio llamado procedimiento remoto (rpc). Esto se logra fácilmente configurando el archivo ejecutable “/etc/rc.d/rc.rpc” y diciéndole que comience. Una vez que se haya configurado como ejecutable, se ejecutará automáticamente cada vez que inicie Slackware.

```
darkstar:~# chmod +x /etc/rc.d/rc.rpc
darkstar:~# /etc/rc.d/rc.rpc start
```

Montar un recurso compartido NFS es un poco diferente a montar un sistema de archivos local. En lugar de especificar un dispositivo local, debe indicar a montar el nombre de dominio o la dirección IP del servidor NFS y el directorio para montar con dos puntos entre ellos.

```
darkstar:~# mount -t nfs darkstar.example.com:/home /home
```

Ejecutar un servidor NFS es un poco diferente. Primero, debe configurar cada directorio para exportar en el archivo `/etc/exports`. “exports” (5) contiene información sobre lo que los directorios se compartirán, con quién se compartirán y qué permisos especiales se otorgarán o denegaran.

```
# See exports(5) for a description.
# This file contains a list of all directories exported to other computers.
# It is used by rpc.nfsd and rpc.mountd.

/home/backup    192.168.1.0/24(sync,rw,no_root_squash)
```

La primera columna en `exports` contiene una lista de los archivos a exportar a través de NFS. La segunda columna es una lista de los sistemas que pueden acceder a la exportación junto con permisos especiales. Puede especificar los hosts a través del nombre de dominio, la dirección IP o la dirección de netblock (como lo he hecho aquí). Los permisos especiales son siempre una lista parentética. Para obtener una lista completa, deberá leer la página de manual. Por ahora, la única

opción especial que importa es *no_root_squash*. Por lo general, el usuario root en un cliente NFS no puede leer ni escribir un recurso compartido exportado. En su lugar, el usuario root es *"aplastado"* y está obligado a actuar como el usuario nobody. *no_root_squash* evita esto.

También necesitarás ejecutar el demonio NFS. El inicio y la detención del soporte del servidor NFS se realiza con el script rc `/etc/rc.d/rc.nfsd` . Establézcalo como ejecutable y ejecútelo como lo hicimos para "rc.rpc" y estará listo para comenzar.

SMB

SMB es el protocolo de intercambio de archivos de la red de Windows. Conectarse a los recursos compartidos de SMB (comúnmente llamadas recursos compartidos de samba) es bastante sencillo. Desafortunadamente, SMB no es tan fuertemente apoyado como NFS. Aún así, ofrece un mayor rendimiento y conectividad con las computadoras con Windows. Por estas razones, SMB es el protocolo común de intercambio de archivos de red implementado en redes locales. La exportación de recursos compartidos SMB desde Slackware se realiza a través del demonio samba y se configura en `smb.conf` (5). Desafortunadamente, configurar samba como un servicio está más allá del alcance de este libro. Consulte en línea para obtener documentación adicional y, como siempre, consulte la página del manual.

Afortunadamente, montar un recurso compartido SMB es fácil y funciona casi exactamente como montar un recurso compartido NFS. Debe indicar a mount dónde encontrar el servidor y a qué recurso compartido desea acceder exactamente de la misma manera. Además, debe especificar un nombre de usuario y contraseña.

```
darkstar:~# mount -t cifs //darkstar/home /home -o
username=alan,password=secret
```

Quizás se esté preguntando por qué el tipo de sistema de archivos es cifs en lugar de smbfs. En versiones anteriores del kernel de Linux, se usaba smbfs. Esto ha quedado en desuso en favor del controlador cifs de propósito general más seguro y con mejor rendimiento.

Todos los recursos compartidos SMB requieren los argumentos *username* y *password* . Esto puede crear un problema de seguridad si desea colocar su recurso de samba en fstab. Puede evitar este problema utilizando el argumento *credentials* . *credentials* apunta a un archivo que contiene la información de nombre de usuario y contraseña. Siempre que este archivo esté protegido y sea legible de forma segura solo por root, se reducirá la posibilidad de que sus credenciales de autenticación se vean comprometidas.

```
darkstar:~# echo "username=alan" > /etc/creds-home
darkstar:~# echo "password=secret" >> /etc/creds-home
darkstar:~# mount -t cifs //darkstar/home -o credentials=/etc/creds-home
```

Navegación de capítulos

Capítulo anterior: Permisos del sistema de archivos

Capítulo siguiente: vi

Sources

- Original source: http://www.slackbook.org/beta
- Originally written by Alan Hicks, Chris Lumens, David Cantrell, Logan Johnson

slackbook, filesystem, network filesystems, nfs, smb, mount