# El intérprete de comandos (shell)

Así que has instalado Slackware y estás mirando un indicador de terminal, ¿y ahora qué? Ahora sería un buen momento para aprender sobre las herramientas básicas de línea de comandos. Y como está mirando un cursor parpadeante, probablemente necesite un poco de ayuda para saber cómo moverse, y de eso se trata este capítulo.

## Documentación del sistema

El sistema Slackware Linux viene con una gran cantidad de documentación incorporada para casi todas las aplicaciones instaladas. Quizás el método más común para leer la documentación del sistema es ***man*** (1). ***man*** (abreviatura de **manual** ) abrirá la página de manual incluida para cualquier aplicación, llamada al sistema, archivo de configuración o biblioteca que le indique. Por ejemplo, **man man** mostrará la página del manual para ***man***.

Desafortunadamente, es posible que no siempre sepa qué aplicación necesita utilizar para la tarea en cuestión. Afortunadamente, ***man*** tiene capacidades de búsqueda integradas. El uso del comodín *-k* hará que ***man*** busque todas las páginas de manual que coincidan con sus términos de búsqueda.

Las páginas del manual están organizadas en grupos o secciones por su tipo de contenido. Por ejemplo, la sección 1 es para aplicaciones de usuario. ***man*** buscará cada sección en orden y mostrará la primera coincidencia que encuentre. A veces encontrará que existe una página de manual en más de una sección para una entrada determinada. En ese caso, deberá especificar la sección exacta para buscar. En este libro, todas las aplicaciones y una serie de otras cosas tendrán un número en su lado derecho entre paréntesis. Este número es la sección de la página de manual donde encontrará información sobre esa herramienta.

```
darkstar:~$ man -k printf
printf               (1)  - format and print data
printf               (3)  - formatted output conversion
darkstar:~$ man 3 printf
```

Secciones de la página de man

| Sección | Contenido |
|---|---|
| 1 | Comandos de Usuario |
| 2 | Llamadas de sistema |
| 3 | Llamadas de bibliotecas C |
| 4 | Dispositivos |
| 5 | Formatos de archivo / Protocolos |
| 6 | Juegos |
| 7 | Convenciones / Paquetes macro |
| 8 | Administración del Sistema |
| 9 | Descripciones API del Kernel |
| n | “New” - Normalmente se utiliza para Tcl/Tk |

# Tratando con archivos y directorios

## Listar archivos y contenidos de los directorios

***ls*** (1) se usa para listar archivos y directorios, sus permisos, tamaño, tipo, número de inodo, propietario y grupo, y mucha información adicional. Por ejemplo, listemos lo que está en el directorio / para su nuevo sistema Slackware Linux.

```
darkstar:~$ ls /
bin/   dev/  home/  lost+found/  mnt/  proc/  sbin/  sys/  usr/
boot/  etc/  lib/   media/       opt/  root/  srv/   tmp/  var/
```

Tenga en cuenta que cada uno de lo listado es un directorio. Estos se distinguen fácilmente de los archivos regulares debido a que finalizan con / (slash); los archivos estándar no tienen ese sufijo. Además, los archivos ejecutables tendrán un sufijo de asterisco. Pero ***ls*** puede hacer mucho más. Para obtener una vista de los permisos de un archivo o directorio, usted necesitará realizar un *"listado largo"*.

```
darkstar:~$ ls -l /home/alan/Desktop
-rw-r--r-- 1 alan users 15624161 2007-09-21 13:02 9780596510480.pdf
-rw-r--r-- 1 alan users  3829534 2007-09-14 12:56 imgscan.zip
drwxr-xr-x 3 alan root       168 2007-09-17 21:01 ipod_hack/
drwxr-xr-x 2 alan users      200 2007-12-03 22:11 libgpod/
drwxr-xr-x 2 alan users      136 2007-09-30 03:16 playground/
```

Una lista larga le permite ver los permisos, el propietario y el grupo, el tamaño del archivo, la fecha de la última modificación y, por supuesto, el nombre del archivo. Observe que las dos primeras entradas son archivos y las tres últimas son directorios. Esto se denota con el primer carácter en la línea. Los archivos regulares obtienen un *"-"* ; los directorios obtienen una *"d"*. Hay varios otros tipos de archivos con sus propios denominadores. Los enlaces simbólicos, por ejemplo, tendrán un *"l"*.

Por último, le mostraremos cómo listar los archivos punto, o archivos ocultos. A diferencia de otros sistemas operativos como Microsoft Windows, no existe ninguna propiedad especial que diferencie los archivos "ocultos" de los archivos "no ocultos". Un archivo oculto simplemente comienza con un punto. Para mostrar estos archivos junto con todos los demás, solo necesita pasar el argumento *-a* a ***ls***.

```
darkstar:~$ ls -a
.xine/    .xinitrc-backup  .xscreensaver  .xsession-errors  SBo/
.xinitrc  .xinitrc-xfce    .xsession      .xwmconfig/       Shared/
```

También es probable que haya notado que sus archivos y directorios aparecen en diferentes colores. Muchas de las funciones mejoradas de ***ls***, como los colores o los caracteres finales que indican el tipo de archivo, son características especiales del programa ***ls*** que se activan al pasar varios argumentos. Para su comodidad, Slackware configura ***ls*** para usar muchos de estos argumentos opcionales de forma predeterminada. Estos están controlados por las variables de entorno LS_OPTIONS y LS_COLORS. Hablaremos más sobre las variables de entorno en el capítulo 5.

### Moviéndose por el sistema de archivos

***cd*** es el comando que se utiliza para cambiar de directorio. A diferencia de la mayoría de los otros comandos, ***cd*** no es en realidad un programa en si, sino que es un comando interno del shell. Básicamente, eso significa que ***cd*** no tiene su propia página de manual. Tendrá que consultar la documentación de su shell para obtener más detalles sobre ***cd*** que puede usar. En su mayor parte, sin embargo, todos se comportan de la misma manera.

```
darkstar:~$ cd /
darkstar:/$ls
bin/   dev/  home/  lost+found/  mnt/  proc/  sbin/  sys/  usr/
boot/  etc/  lib/   media/       opt/  root/  srv/   tmp/  var/
darkstar:/$cd /usr/local
darkstar:/usr/local$
```

¿Observa cómo cambia el prompt cuando cambiamos de directorio? El shell predeterminado de Slackware hace esto como una forma rápida y fácil de ver su directorio actual, pero en realidad no es una función de ***cd***. Si su shell no funciona de esta manera, puede obtener fácilmente su directorio de trabajo actual con el comando ***pwd*** (1). (La mayoría de los shells de UNIX tienen prompts configurables que pueden ser cambiados para proporcionar esta misma funcionalidad. De hecho, esta es otra configuración de conveniencia en el shell por defecto de Slackware).

```
darkstar:~$ pwd
/usr/local
```

## Creación y eliminación de archivos y directorios

Si bien la mayoría de las aplicaciones pueden crear y crearán sus propios archivos y directorios, a menudo querrá hacer esto por su cuenta. Afortunadamente, es muy fácil usar ***touch***(1) y ***mkdir***(1).

***touch*** en realidad modifica la fecha y hora en un archivo, pero si ese archivo no existe, lo creará.

```
darkstar:~/foo$ ls -l
-rw-r--r-- 1 alan users 0 2012-01-18 15:01 bar1
darkstar:~/foo$ touch bar2
-rw-r--r-- 1 alan users 0 2012-01-18 15:01 bar1
-rw-r--r-- 1 alan users 0 2012-01-18 15:05 bar2
darkstar:~/foo$ touch bar1
-rw-r--r-- 1 alan users 0 2012-01-18 15:05 bar1
-rw-r--r-- 1 alan users 0 2012-01-18 15:05 bar2
```

Observe cómo fue creado `bar2` en nuestro segundo comando, y el tercer comando simplemente actualiza la fecha y hora en `bar1`.

***mkdir*** se utiliza para (obviamente ) crear directorios. mkdir foo creará el directorio “foo” dentro del directorio de trabajo actual. Además, puede usar la opción *-p* para crear cualquier directorio principal faltante.

```
darkstar:~$ mkdir foo
darkstar:~$ mkdir /slack/foo/bar/
mkdir: cannot create directory `/slack/foo/bar/': No such file or directory
darkstar:~$ mkdir -p /slack/foo/bar/
```

En este último caso, ***mkdir*** creará primero "/slack", luego "/slack /foo", y finalmente "/slack/foo/bar". Si no usa el argumento *-p*, ***mkdir*** no podrá crear "/slack/foo/bar" a menos que los dos primeros ya existieran, como se ve en el ejemplo.

Eliminar un archivo es tan fácil como crear uno. El comando ***rm***(1) eliminará un archivo (asumiendo, por supuesto, que tenga permiso para hacerlo). Hay algunas opciones muy comunes para ***rm***. La primera es *-f* y se usa para forzar la eliminación de un archivo que puede carecer de permiso explícito para eliminar. La opción *-r* eliminará los directorios y sus contenidos de forma recursiva.

Hay otra herramienta para eliminar directorios, el humilde ***rmdir*** (1). ***rmdir*** solo eliminará los directorios que estén vacíos y se quejará ruidosamente de aquellos que contienen archivos o subdirectorios.

```
darkstar:~$ ls
foo_1/ foo_2/
darkstar:~$ ls foo_1
bar_1
darkstar:~$ rmdir foo_1
rmdir: foo/: Directory not empty
darkstar:~$ rm foo_1/bar
darkstar:~$ rmdir foo_1
darkstar:~$ ls foo_2
bar_2/
darkstar:~$ rm -fr foo_2
darkstar:~$ ls
```

# Archivar y comprimir

¿Todos necesitan empaquetar muchos archivos pequeños juntos para un fácil almacenamiento de vez en cuando, o quizás necesita comprimir archivos muy grandes en un tamaño más manejable? ¿Tal vez quiere hacer ambas cosas juntas? Afortunadamente hay varias herramientas para hacer precisamente eso.

## zip y unzip

Probablemente esté familiarizado con los archivos .zip. Estos son archivos comprimidos que contienen otros archivos y directorios. Aunque normalmente no usamos estos archivos en el mundo de Linux, todavía son utilizados comúnmente por otros sistemas operativos, por lo que ocasionalmente tenemos que tratar con ellos.

Para crear un archivo zip, (naturalmente) utilizará el comando ***zip*** (1). Puede comprimir archivos o directorios (o ambos) con ***zip***, pero tendrá que usar la opción *-r* para hacerlo en forma recursiva con los directorios.

```
darkstar:~$ zip -r /tmp/home.zip /home
darkstar:~$ zip /tmp/large_file.zip /tmp/large_file
```

El primer nombre debe ser el del archivo zip que se creará (si se omite la extensión .zip, ***zip*** lo agregará por usted) y el resto son archivos o directorios que se agregarán al archivo zip.

Naturalmente, ***unzip*** (1) descomprimirá un archivo comprimido zip.

```
darkstar:~$ unzip /tmp/home.zip
```

## gzip

Una de las herramientas de compresión más antiguas incluidas en Slackware es ***gzip*** (1), una herramienta de compresión que solo es capaz de tomar un solo archivo a la vez. Mientras que ***zip*** es una herramienta de compresión y de archivado, ***gzip*** solo es capaz de comprimir. A primera vista, esto parece una desventaja, pero es realmente una fortaleza. La filosofía de UNIX de hacer pequeñas herramientas que hacen bien sus pequeños trabajos permite que se combinen de muchas maneras. Para comprimir un archivo (o varios archivos), simplemente páselos como argumentos a ***gzip***. Cada vez que ***gzip*** comprime un archivo, agrega una extensión .gz y elimina el archivo original.

```
darkstar:~$ gzip /tmp/large_file
```

Descomprimir es igual de sencillo con ***gunzip*** que creará un nuevo archivo sin comprimir y eliminará el anterior.

```
darkstar:~$ gunzip /tmp/large_file.gz
darkstar:~$ ls /tmp/large_file*
/tmp/large_file
```

Pero supongamos que no queremos eliminar el archivo comprimido anterior, solo queremos leer su contenido o enviarlo como entrada a otro programa. El programa ***zcat*** leerá el archivo gzip, lo descomprimirá en la memoria y enviará el contenido a la salida estándar (la pantalla del terminal, a menos que se redirija, consulte la sección denominado "Redirección de entrada y salida" para obtener más detalles sobre la redirección de salida).

```
darkstar:~$ zcat /tmp/large_file.gz
Wed Aug 26 10:00:38 CDT 2009
Slackware 13.0 x86 is released as stable!  Thanks to everyone who helped
make this release possible -- see the RELEASE_NOTES for the credits.
The ISOs are off to the replicator.  This time it will be a 6 CD-ROM
32-bit set and a dual-sided 32-bit/64-bit x86/x86_64 DVD.  We're taking
pre-orders now at store.slackware.com.  Please consider picking up a copy
to help support the project.  Once again, thanks to the entire Slackware
community for all the help testing and fixing things and offering
suggestions during this development cycle.
```

## bzip2

Una alternativa a ***gzip*** es la utilidad de compresión ***bzip2*** (1) que funciona casi de la misma manera. La ventaja de ***bzip2*** es que cuenta con una mayor compresión. Desafortunadamente, lograr una mayor compresión es un proceso lento y que requiere una gran cantidad de CPU, por lo que ***bzip2*** normalmente demora mucho más en ejecutarse que otras alternativas.

## XZ / LZMA

La última utilidad de compresión agregada a Slackware es ***xz***, que implementa el algoritmo de compresión LZMA. Esto es más rápido que ***bzip2*** y, a menudo, también comprime mejor. De hecho, su combinación de velocidad y fuerza de compresión hizo que reemplazara ***gzip*** como el esquema de compresión elegido por Slackware. Desafortunadamente, ***xz*** no tiene una página de manual al momento de escribir esto, por lo que para ver las opciones disponibles, use el argumento *-help*. La compresión de archivos se realiza con el argumento *-z*, y la descompresión con *-d*.

```
darkstar:~$ xz -z /tmp/large_file
```

## tar

Muy bien, sabemos cómo comprimir archivos utilizando todo tipo de programas, pero ninguno de ellos puede archivar los archivos de la forma en que lo hace ***zip***. Eso es hasta ahora. El Archivador de cinta (**T**ape **ar**chiver), o ***tar*** (1) es el programa de archivado más utilizado en Slackware. Al igual que otros programas de archivo, ***tar*** genera un nuevo archivo que contiene otros archivos y directorios. No comprime el archivo generado (a menudo llamado “tarball”) de forma predeterminada; sin embargo, la versión de ***tar*** incluida en Slackware es compatible con una variedad de esquemas de compresión, incluidos los mencionados anteriormente. Invocar ***tar*** puede ser tan fácil o tan complicado como quieras. Normalmente, la creación de un archivo tar se realiza con los argumentos - *cvzf* . Echemos un vistazo a estos en profundidad.

tar Opciones

| Opción | Qué hace |
|---|---|
| c | Crea un tarball |
| x | Extrae el contenido de un tarball |
| t | Muestra el contenido de untarball |
| v | Muestra los archivos que están siendo desarchivados |
| z | Usa compresión gzip |
| j | Usa compresión bzip2 |
| J | Usa compresión LZMA |
| p | Conserva permisos |

***tar*** requiere un poco más de precisión que otras aplicaciones en el orden de sus argumentos. El argumento - *f* debe estar presente al leer o escribir en un archivo, por ejemplo, y lo siguiente que debe seguir es el nombre del archivo. Considera los siguientes ejemplos.

```
darkstar:~$ tar -xvzf /tmp/tarball.tar.gz
```

```
darkstar:~$ tar -xvfz /tmp/tarball.tar.gz
```

Arriba, el primer ejemplo funciona como se espera, pero el segundo falla porque ***tar*** recibió instrucciones de abrir el archivo `z` en lugar del `/tmp/tarball.tar esperado. gz` .

Ahora que hemos resuelto las opciones, veamos algunos ejemplos de cómo crear y extraer tarballs. Como hemos señalado, la opción - *c* se utiliza para crear archivos comprimidos y - *x* extrae su contenido. Sin embargo, si queremos crear o extraer un tarball comprimido, también tenemos que especificar la compresión adecuada para usar. Naturalmente, si no queremos comprimir el tarball, podemos dejar de lado estas opciones. El siguiente comando crea un nuevo archivo comprimido usando el algoritmo de compresión ***gzip*** . Si bien no es un requisito estricto, también es una buena práctica agregar la extensión .tar a todos los archivos comprimidos, así como a cualquier extensión utilizada por el algoritmo de compresión.

```
darkstar:~$ tar -czf /tmp/tarball.tar.gz /tmp/tarball/
```

# Lectura de documentos

Tradicionalmente, los sistemas operativos UNIX y similares a UNIX están llenos de archivos de texto que en algún momento los usuarios del sistema van a querer leer. Naturalmente, hay muchas maneras de leer estos archivos, y le mostraremos las más comunes.

En los primeros días, si solo quería ver el contenido de un archivo (cualquier archivo, ya sea un archivo de texto o algún programa binario) usaría ***cat*** (1) para verlos. ***cat*** es un programa muy simple, que toma uno o más archivos, los concatena (de ahí el nombre) y los envía a la salida estándar, que suele ser la pantalla de su terminal. Esto estaba bien cuando el archivo era pequeño y no se desplazaba fuera de la pantalla, pero era inadecuado para archivos más grandes, ya que no tenía una forma integrada de moverse dentro de un documento y leer un párrafo a la vez. Hoy en día, ***cat*** todavía se usa bastante, pero predominantemente en scripts o para unir dos o más archivos en uno.

```
darkstar:~$ cat /etc/slackware-version
Slackware 14.0
```

Dadas las limitaciones de ***cat*** algunas personas muy inteligentes se sentaron y comenzaron a trabajar en una aplicación para permitirles leer documentos una página a la vez. Naturalmente, tales aplicaciones comenzaron a ser conocidas como “paginadores”. Uno de los primeros fue ***more*** (1), nombrado porque le permitiría ver “más” del archivo cuando lo deseara.

## more

***more*** mostrará las primeras líneas de un archivo de texto hasta que la pantalla esté llena, luego se detendrá. Una vez que haya leído esa pantalla, puede avanzar una línea presionando la tecla ENTER, o una pantalla completa presionando SPACE, o un número específico de líneas escribiendo un número y luego la barra SPACE. ***more*** también es capaz de buscar palabras clave en un archivo de texto; una vez que haya mostrado un archivo en ***more***, presione la tecla / e ingrese una palabra clave. Al presionar ENTER, el texto se desplazará hasta que encuentre la próxima coincidencia.

Esto es claramente una gran mejora con respecto a ***cat***, pero todavía sufre de algunas fallas molestas; ***more*** no puede desplazarse hacia atrás a través de un archivo de una tubería (piped) para permitirle leer algo que podría haber perdido, la función de búsqueda no resalta sus resultados, no hay desplazamiento horizontal, etc. Claramente es posible una mejor solución.

De hecho, las versiones modernas de ***more***, como la que se incluye con Slackware, presentan una función **back** a través de la tecla b. Sin embargo, la función solo está disponible al abrir archivos directamente en ***more***; no cuando un archivo llega desde una tubería a ***more***.

### less

Para solucionar los inconvenientes de ***more***, se desarrolló un nuevo paginador que, irónicamente, se denominó ***less*** (1). ***less*** es un paginador muy poderoso que soporta todas las funciones de ***more*** mientras agrega muchas funciones adicionales. Para empezar, ***less*** le permite usar las teclas de flecha para controlar el movimiento dentro del documento.

Debido a su popularidad, muchas distribuciones de Linux han comenzado a excluir ***more*** a favor de ***less***. Slackware incluye ambos. Además, Slackware también incluye un pequeño preprocesador para ***less*** llamado `lesspipe.sh` . Esto permite a un usuario ejecutar ***less*** en una serie de archivos que no son de texto. `lesspipe.sh` generará un resultado de texto al ejecutar un comando en estos archivos y lo mostrará en ***less***.

***less*** proporciona casi tanta funcionalidad como se podría esperar de un editor de texto sin ser realmente un editor de texto. El movimiento línea por línea que se puede hacer en ***vi*** - como j y k, o con las teclas de flecha, oENTER. En el caso de que un archivo sea demasiado ancho para caber en una pantalla, incluso puede desplazarse horizontalmente con las teclas de flecha izquierda y derecha. La tecla g lo lleva al principio del archivo, mientras que la tecla G lo lleva al final.

La búsqueda se realiza como con ***more***, escribiendo la tecla / y luego su cadena de búsqueda, pero observe cómo se resaltan los resultados de la búsqueda y al escribir n lo llevará a la siguiente aparición del resultado, mientras que N lo llevará a la aparición anterior.

Igual que con ***more***, los archivos pueden abrirse directamente en ***less*** o canalizarse a este:

```
darkstar:~$  less
/usr/doc/less:/README
darkstar:~$  cat
/usr/doc/less:/README
/usr/doc/util-linux:/README | less
```

Hay mucho más para ***less***; desde la aplicación, escriba h para obtener una lista completa de comandos.

## Enlaces

Los enlaces son un método para referirse a un archivo por más de un nombre. Al utilizar la aplicación ***ln*** (1), un usuario puede hacer referencia a un archivo con más de un nombre. Los dos archivos no son copias iguales de uno a otro, sino que son exactamente el mismo archivo, solo que con un

nombre diferente. Para eliminar el archivo por completo, todos sus nombres deben ser eliminados. (Esto es en realidad el resultado de la forma en que funcionan ***rm*** y otras herramientas similares. En lugar de eliminar el contenido del archivo, simplemente eliminan la referencia al archivo, liberando ese espacio para volver a utilizarlo. ***ln*** creará una segunda referencia o "enlace" a ese archivo).

```
darkstar:~$ ln /etc/slackware-version foo
darkstar:~$ cat foo
Slackware 14.0
darkstar:~$ ls -l /etc/slackware-version foo
-rw-r--r-- 1 root root 17 2007-06-10 02:23 /etc/slackware-version
-rw-r--r-- 1 root root 17 2007-06-10 02:23 foo
```

Otro tipo de enlace existe, el enlace simbólico. Los enlaces simbólicos, en lugar de ser otra referencia al mismo archivo, son en realidad un tipo especial de archivo por derecho propio. Estos enlaces simbólicos apuntan a otro archivo o directorio. La principal ventaja de los enlaces simbólicos es que pueden referirse a directorios así como a archivos, y pueden abarcar múltiples sistemas de archivos. Estos se crean con el argumento *-s*.

```
darkstar:~$ ln -s /etc/slackware-version foo
darkstar:~$ cat foo
Slackware 140
darkstar:~$ ls -l /etc/slackware-version foo
-rw-r--r-- 1 root root 17 2007-06-10 02:23 /etc/slackware-version
lrwxrwxrwx 1 root root 22 2008-01-25 04:16 foo -> /etc/slackware-version
```

Cuando utilice enlaces simbólicos, recuerde que si se elimina el archivo original, su enlace simbólico no sirve para nada; simplemente apunta a un archivo que ya no existe.

# Navegación de capítulos

**Capítulo previo: Arranque**

**Próximo capítulo: The Bourne Again Shell**

# Fuentes

- Fuente Original: http://www.slackbook.org/beta

- Escrito originalmente por Alan Hicks, Chris Lumens, David Cantrell, Logan Johnson
- Traducido por: Victor 2019/02/01 18:44

slackbook, shell, archive, filesystem