Proyecto de Servicio Social Apoyo a los Servicios Estudiantiles y Académicos:
Asesorías para Estudiantes en Aulas y Talleres de Ciencias de la Computación.
No. Programa: 2022-12/12-1273
Autor: Miriam Torres Bucio.

Comandos básicos de Linux

Objetivo: Este documento está diseñado para que los alumnos de la Facultad de Ciencias de la UNAM puedan tener un listado de los comandos básicos de Linux.

Los comandos en la consola son de utilidad al momento de resolver algún problema en la computadora, así como realizar algunas actividades por ejemplo: crear carpetas, acceder a directorios, etc. Por lo tanto, es material de utilidad para poder apoyarse en ellos y conocer el Sistema Linux.

Índice

1. ¿Qué es un comando en Linux?

Un comando Linux es un programa o utilidad que se ejecuta en la línea de comandos. Una línea de comandos es una interfaz que acepta líneas de texto y las procesa en forma de instrucciones para tu ordenador.

Hay que tener en cuenta que en cualquier interfaz gráfica de usuario ocurren lineas de comando pues no son otra cosa más que una abstracción de los programas de línea de comandos.
Por ejemplo, cuando cierramos una ventana haciendo clic en la 'X' de la ventanilla en la parte superior ya sea izquierda o derecha, hay un comando que se ejecuta detrás de esa acción.

Ahora bien, para poder ejecutar los siguientes comandos es necesario hacer uso de la Consola de Linux y así realizar diferentes acciones. Esta la podemos obtener pulsando Ctrl+Alt+T en nuestro teclado.

1. pwd: Nos da una ruta absoluta es decir, la ruta del directorio en el que estamos trabajando.
2. cd : Podemos acceder a las carpetas de nuestra elección. De igual maera existen otros comandos como son:
   - cd ..: Nos regresa a una carpeta anterior.
   - cd: Nos regresa a la carpeta raíz o de inicio.
3. ls: Nos da un listado de los documentos que tenemos en una carpeta.
4. cat: Podemos crear un archivo desde cero con este comando o bien darnos el contenido de un archivo.
5. cp: Copia un archivo de un directorio a otro, basta con escribir el archivo que se quiere copiar seguido del archivo o directorio de destino.
   - cp -r: Copia directorios enteros.
6. mv: Mueve archivos de un directorio a otro.
7. mkdir: Crea carpetas dentro de un directorio.
   - mkdir -p: Crea subdirectorios.
8. rmdir: Elimina directorios vacíos.
9. rm: Elimina directorios. ¡Cuidado! No se pueden recuperar los archivos después de esta acción.
   - rm -r: Elimina un directorio vacío.
   - rm -rf: Elimina directorios con elementos dentro de él.
10. locate: Localiza archivos.
11. grep: Localiza palabras en un texto.
    - grep -c palabra: Cuenta el número de veces que aparece la palabra.
12. sudo: Este comando le permite realizar tareas que requieren permisos administrativos o de raíz. Sin embargo, no es recomendable usar este comando para el uso diario porque puede ser fácil que ocurra un error si hizo algo mal.
13. df: Obtiene un informe sobre el uso del espacio en disco del sistema, que se muestra en porcentaje y KB. En megabytes, es con df -m .
14. du: Nos da la cantidad de espacio en memoria que ocupa un archivo o un directorio. Si deseamos verlo en bytes, kilobytes y megabytes, agregamos el argumento -h a la línea de comando.

15. head: Se utiliza para ver las primeras líneas de un archivo de texto.
    - tail -n 4: Vemos las primeras tres líneas.
16. tail: Análogamente a head, tail muestra las ultimas líneas de un archivo de texto.
    - tail -n 3: Vemos las últimas tres líneas.
17. man: Muestra la página del manual de cualquier comando. Por ejemplo, si ejecutamos man mkdir nos mostrará el manual del comando mkdir.
18. touch: Permite actualizar los tiempos de acceso y modificación de los archivos especificados.

    No obstante, la mayoría de las veces no se utiliza touch para modificar las fechas de los archivos, sino para crear nuevos archivos vacíos por ejemplo touch new_file
19. exit: Permite salir del shell.
20. ./: Permite al shell ejecutar un archivo que sea ejecutable con algún intérprete que se tenga instalado en el sistema directamente en la terminal.

    Un pequeño detalle al usar este comando es que se deben tener los permisos de ejecución del archivo.
21. sudo: Este comando significa *superuser do* el cual nos permite actuar como superusuario o usuario root mientras ejecutamos algún comando en específico.

    Es la forma en que Linux se protege a sí mismo y evita que los usuarios modifiquen accidentalmente el sistema de archivos de la máquina o instalen paquetes inapropiados.

    Sudo se utiliza para la instalación o actualización de paquetes del sistema.
22. shutdown: Permite apagar, detener o reiniciar la computadora.
    - shutdown now: Apaga eordenador de manera inmediata.
    - shutdown 21:30: Programa el apagado de la computadora.
    - shutdown -c: Cancela una llamada anterior.
23. htop: Es un visualizador de procesos interactivo que permite gestionar los recursos de la computadora.
24. unzip: Extrae el contenido de un archivo .zip desde la terminal.
25. apt, yum, pacman: Son gestores de paquetes para hacer instalaciones.
    - apt: Basado en Debian(Ubuntu, LinuxMint)
      sudo apt install package_name
    - yum: Basado en Red Hat(Fedora, CentOS)
      sudo yum install package_name
    - pacman: Basado en Arch(Manjaro, Arco Linux)
      sudo pacman -S package_name
26. echo: Muestra el texto definido en el terminal. Su uso principal es imprimir las variables de entorno dentro de esos mensajes.
27. ps: Podemos ver los procesos que están sucediendo en el shell que se están ejecutando.

    Imprime información útil sobre los programas que está ejecutando, como el ID del proceso, el TTY (Teletipo), la hora y el nombre del comando.

28. kill: Termina un proceso. Es útil cuando un programa que se está ejecutando no termina o no responde.
    Hay que tener cuidado al usarlo pues se corre el riesgo de borrar accidentalmente el trabajo ya hecho.
29. ping: Se utiliza para verificar la conectividad a la red. Se utiliza para verificar la dirección IP de un sitio.
30. vim: Es un editor de texto libre directo de la terminal.
31. history: Muestra un listado de comandos que se han utilizado es útil pues si no recordamos alguno lo muestra mediante una lista.
32. which: Muestra la ruta completa de los comandos del shell.
33. shred: Este comando anula el contenido de un archivo repetidamente, y como resultado, el archivo dado se vuelve extremadamente difícil de recuperar.
    - shred -u: Elimina de inmediato dicho archivo.
34. less: Permite inspeccionar archivos hacia atrás y hacia adelante.
35. whoami: Muestra el nombre de usuario con el cual estamos trabajando.
36. whatis: Imprime una descripción de una sola línea de cualquier otro comando.
37. wc: Devuelve el número de líneas, palabras, tamaño y nombre de un archivo de texto.
    - wc -w: Devuelve solo el número de palabras en el archivo de texto.
38. uname: Imprime la información del sistema operativo, lo que resulta útil cuando se conoce la versión actual de Linux.
    - uname -a: Dice toda la información del sistema.
39. neofetch: Muestra información sobre el sistema, así como la versión del kernel, el shell y el hardware junto a un logotipo ASCII de la distribución Linux que se está usando.
40. find: Busca archivos en una jerarquía de directorios.
    Para usarlo es mediante la siguiente sintaxis: find [flags] [path] -name [expression]
41. df -h: Nos dice las particiones que tenemos en nuestro disco duro.
42. fsck: Nos ayuda a resolver problemas que podemos llegar a tener en el disco duro y sus particiones.
    Para mas informacion podemos ejecutar:

    ```
    fsck --help
    ```

43. ntfsfix: Sirve para reparar un sistema de archivos NTFS.
44. cfdisk: Es de utilidad para crear, editar o eliminar particiones de un disco duro de manera segura.
45. blkid: Es utilizado para brindar información de bloques disponibles en un disco duro.
    Determina el tipo de contenido del disco por ejemplo, el sistema de archivos que tiene un dispositivo de bloque así como, atributosde los metadatos y su contenido como los campos LABEL o UUID.

    Para más información puedes consultar el siguiente link. [1]
46. mokutil –sb -state: Nos dice si SecureBoot esta habilitado o deshabilitado.

[1] L. (2022a, junio 21). Blkid Command in Linux to Find Block Devices Details. LinuxOPsys. Recuperado 13 de marzo de 2023, de https://linuxopsys.com/topics/blkid-command-in-linux

2. Comandos en Fedora

En Linux Ubuntu por ejemplo, si queremos instalar java se hace mediante la siguiente línea de comandos:

```
$ sudo apt install default-jre
```

Caso que no ocurre con Fedora. Para poder hacer la instalación de Java en Fedora (versiones actuales 34, 35 o 36) el comando es un poco diferente:

```
$ sudo yum install -y java-11-openjdk
```

Fedora utiliza la palabra reservada yum para poder ejecutar comandos. Sin embargo, en versiones más actuales se utiliza dnf en lugar de *yum*.

La siguiente es una lista de ejemplos de algunos comandos en Fedora:

1. dnf -y update: Actualiza el sistema.
2. dnf remove: Elimina un paquete.

3. Comando chmod

chmod es un comando que nos permite editar el permiso de archivos tanto de escritura, lectura y ejecución. Para poder hacer los cambios en los archivos, es necesario ser administrador es decir, contar con los derechos raíz o ser dueño del archivo.
Los permisos de archivos suelen aparecer con las siguientes letras las cuales son sus abreviaciones:

- r : leer

  Leer el contenido del directorio.

- w : escribir

  Crear, modificar y borrar archivos en el directorio.

- x : ejecutar / chdir

  Cambiarse al directorio.

Ahora bien, existen tres columnas de permisos en Unix y cada columna aplica a un sujeto diferente.
De manera gráfica, los permisos y 'sujetos' se ven así:

```
rwx   rwx   rwx
 u     g     o
```

Los sujetos son las abreviaciones:

- u : usuario
- g : grupo
- o : otro
- a : todos los anteriores

Ahora, para ver los permisos de archivos tenemos que estar situados en el archivo o directorio y usamos el comando ls -l.

Lo que nos aparecerá en consola es algo similar a lo que se muestra a continuación:

```
total 2
-rw-rw-r-- 1 det det 45 ene 06 10:45 tarea1.pdf
-rw-rw-r-- 1 det det 98 ene 06 10:45 practica1.java
```

Una observación que hay que hacer es que el primer caracter de los permisos tienen un significado.

- - : Archivo común y corriente.
- d : Indica que estamos sobre un directorio.
- otros caracteres : Pueden indicar tipos de archivos especiales.

Por otro lado la palabra det se refiere al nombre de usuario y el siguiente bloque con números es el tamaño del archivo.

En el ejemplo al inicio de cada línea que nos dice los permisos del archivo aparece un {-} lo que quiere decir que se trata de los permisos de un archivo común, en este caso de un pdf.

Por otro lado, podemos agrupar esa información de la siguiente forma:

```
- rw- rw- r--
  u   g   o
```

Podemos deducir que este permiso de archivos se trata de un archivo común y corriente (es el caso de tarea1.pdf) en donde:

- El usuario tiene permiso de lectura y escritura.
  El usuario NO tiene permiso de ejecución.
- Un grupo tiene permiso de lectura y escritura.
  El grupo NO tiene permiso de ejecución.
- Otros solo tienen permiso de lectura.
  Otros NO tienen permiso de escritura ni de ejecución.

3.1. Agregar o quitar permisos en archivos.

Podemos cambiar los permisos de los archivos para que las tres columnas de permisos tengan acceso a un archivo o para otorgar permisos específicos o para que nadie tenga acceso a un archivo. Sin embargo, hay que tener cuidado al otorgar o quitar permisos pues puede ser contraproducente.

Por ejemplo, una manera de otorgar el permiso al usuario de ejecución en un archivo es con el comando
chmod u+x name_file

Si usamos ese comando en el archivo tarea1.pdf los permisos cambiarían y de esta forma el usuario tendría el permiso de ejecutar el archivo.
chmod u+x tarea1.pdf

```
- rwx rw- r-- 1 det det 45 ene 06 11:31 tarea1.pdf
  u   g   o
```

Otro ejemplo sería otorgar permisos a otros para poder leer y ejecutar el archivo.
Para esto usamos chmod o+wx tarea1.pdf

```
- rwx rw- rwx 1 det det 45 ene 06 11:31 tarea1.pdf
  u   g   o
```

Finalmente, para poder quitar permisos en los archivos lo podemos hacer sustituyendo el caracter {+} del comando que hemos estado usando por el comando {-}.

Por ejemplo, para dejar la columna de otros como lo teníamos hay que quitar los permisos de escritura y ejecución, entonces usamos la línea de comandos chmod o-wx tarea1.pdf

```
- rwx rw- r-- 1 det det 45 ene 06 11:40 tarea1.pdf
  u   g   o
```

Como dato adicional, por ejemplo si queremos agregar el permiso de ejecución a las tres columnas podemos hacerlo con el comando chmod a+x name_file en donde 'a' significa 'para todos'.

Para el archivo pdf con el que hemos dado los ejemplos, el comando sería chmod a+x tarea1.pdf

```
- rwx rwx r-x 1 det det 45 ene 06 11:51 tarea1.pdf
  u   g   o
```

La mayoría de los comandos de Linux que vimos fueron consultados en el siguiente link. [2]

[2] A., D. (2023, 10 marzo). 40 Comandos básicos de Linux que todo usuario debe saber. Tutoriales Hostinger. Recuperado 14 de diciembre de 2022, de https://www.hostinger.mx/tutoriales/linux-comandos