

Universidad Tecnológica de Panamá
Sistemas Operativos I
Experiencia Práctica en Laboratorio No. 3
Manejo de Archivos

Prof. Aris Castillo de Valencia

Objetivo:

- Probar y distinguir distintos comandos para trabajar con archivos en línea de texto, incluyendo: moverse entre carpetas, buscar información dentro de archivos de texto, agregar texto, desplegar en pantalla el contenido de archivos y realizar copias de seguridad.

Metas:

Con esta experiencia práctica se espera que el estudiante sea capaz de realizar tareas sencillas de administración del sistema operativo Linux/GNU a través de comandos para manipular archivos.

Contenidos:

- Comandos de manejo de archivos: cd, cat, less, more, head, tail, find, grep, ls, file, cpio.

Metodología:

Se basa en métodos intuitivos, de experimentación y demostración en que se acerca al estudiante a situaciones reales de la práctica profesional de manera que resuelva las situaciones presentadas.

Evaluación:

- Se dará 50 puntos por el desarrollo de la práctica en el aula.
- Se dará 50 puntos por la entrega del informe escrito debidamente completado y por su nivel técnico.

Recursos:

- Hardware: computadora, conexión a Internet.

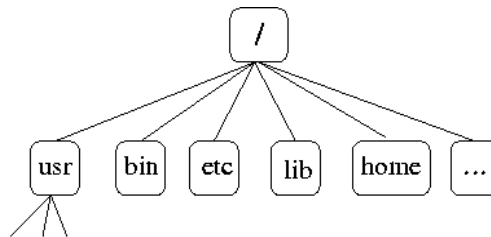
- Software: Sistema operativo Linux/GNU.

Procedimiento:

Lea cuidadosamente la guía; pruebe cada uno de los comandos listados prestando especial atención a los resultados obtenidos y a las variantes que le ofrecen las opciones de los comandos. Ponga en práctica los comandos aprendidos haciendo los ejercicios sugeridos. Llene la autoevaluación y retroalimentación y súbala a la plataforma Moodle.

¿Cómo me muevo entre carpetas?

El comando `cd` (change directory) le permite cambiar de directorio. Son muchas las opciones que puede utilizar dependiendo de lo que desea realizar. Trabajaremos con el árbol de directorio de Linux, que es clave para su comprensión de este sistema operativo.



Recuerde que Linux es un sistema operativo **multiusuario**. Para cada usuario creado, el sistema le crea una carpeta de trabajo, comúnmente llamada **“home,”** las cuales estarán ubicadas dentro del directorio **“home.”** Tenga presente que, generalmente, cuando se habla de ir a **“home”** no se trata de la carpeta **“home”** dentro del directorio raíz **“/”** sino de *la carpeta del usuario actualmente en sesión*. Por ejemplo, si usted es el usuario X, habrá una carpeta llamada X, que contendrá todos sus archivos y directorios. Si existe otro usuario Y en el sistema, habrá otra carpeta Y, que contendrá todos los archivos y directorios del usuario Y. Así sucesivamente ocurrirá para cada usuario existente. Eso es lo que comúnmente se le llama **“home.”**

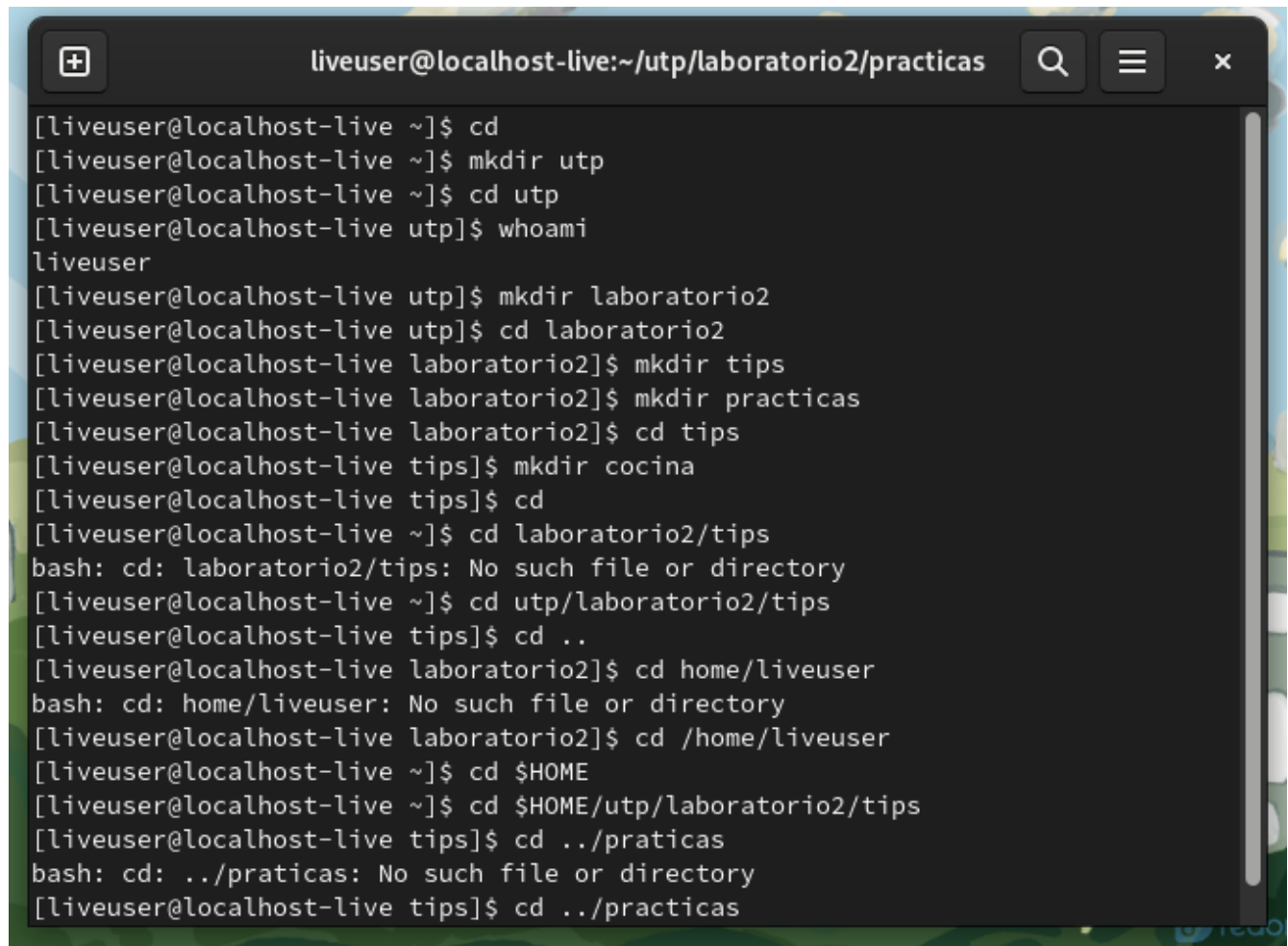
Sintaxis: **`cd nombre_directorio`**

Resultado: se pasa a la carpeta nombre_directorio

Algunos ejemplos:

Antes de realizar los ejemplos, cree una carpeta llamada laboratorio2. Dentro de ésta cree dos subcarpetas – tips y practicas. Dentro de tips cree una carpeta llamada cocina.

| Comando ejecutado | Resultado |
|------------------------------------|---|
| cd laboratorio2 | Se pasa del directorio home del usuario actualmente en sesión a la carpeta laboratorio2. Note que Linux le coloca el nombre del directorio donde se encuentra actualmente. |
| cd | Permite regresar al directorio home del usuario actualmente en sesión |
| cd laboratorio2/tips | Se pasa en un sólo paso del directorio home del usuario actualmente en sesión a la carpeta tips que se encuentra dentro de la carpeta laboratorio2 |
| cd .. | Se mueve hacia atrás, es decir, al directorio padre del directorio actual. En este caso como la carpeta tips se encontraba dentro de la carpeta laboratorio2, Linux lo deja en ésta última. |
| cd /home/aris | Se mueve al directorio home de mi usuario. En su caso, reemplace aris por el nombre de su usuario. Recuerde que para moverse al directorio home de su usuario, no es necesario colocar toda la ruta, sino solo ejecutar cd. Esto sucede pues el directorio por default es el del usuario en sesión. |
| cd \$HOME | Hace lo mismo que la secuencia anterior. Se mueve al directorio home del usuario en sesión. En este caso \$HOME es una variable de ambiente. |
| cd \$HOME/laboratorio2/tips | Lo lleva al directorio tips. Cuál sería la forma utilizando la ruta completa? |
| cd ../practicas | Lo lleva al directorio practicas que está dentro de laboratorio2. En este caso ../ está repitiendo la ruta del directorio padre del directorio tips donde nos encontrábamos antes de ejecutar el comando. Cuál sería la otra alternativa? |



```
liveuser@localhost-live:~/utp/laboratorio2/practicas
[liveuser@localhost-live ~]$ cd
[liveuser@localhost-live ~]$ mkdir utp
[liveuser@localhost-live ~]$ cd utp
[liveuser@localhost-live utp]$ whoami
liveuser
[liveuser@localhost-live utp]$ mkdir laboratorio2
[liveuser@localhost-live utp]$ cd laboratorio2
[liveuser@localhost-live laboratorio2]$ mkdir tips
[liveuser@localhost-live laboratorio2]$ mkdir practicas
[liveuser@localhost-live laboratorio2]$ cd tips
[liveuser@localhost-live tips]$ mkdir cocina
[liveuser@localhost-live tips]$ cd
[liveuser@localhost-live ~]$ cd laboratorio2/tips
bash: cd: laboratorio2/tips: No such file or directory
[liveuser@localhost-live ~]$ cd utp/laboratorio2/tips
[liveuser@localhost-live tips]$ cd ..
[liveuser@localhost-live laboratorio2]$ cd home/liveuser
bash: cd: home/liveuser: No such file or directory
[liveuser@localhost-live laboratorio2]$ cd /home/liveuser
[liveuser@localhost-live ~]$ cd $HOME
[liveuser@localhost-live ~]$ cd $HOME/utp/laboratorio2/tips
[liveuser@localhost-live tips]$ cd ../practicas
bash: cd: ../practicas: No such file or directory
[liveuser@localhost-live tips]$ cd ../practicas
```

- Pase al directorio raíz. Luego liste su contenido.

```
[liveuser@localhost-live practicas]$ cd
[liveuser@localhost-live ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  utp  Videos
```

- Muévase al directorio /home. Liste su contenido.

```
[liveuser@localhost-live ~]$ cd /home/liveuser
[liveuser@localhost-live ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  utp  Videos
```

- Regrese al directorio home de su usuario. Liste la secuencia de comandos ejecutados de forma completa.

```
[liveuser@localhost-live practicas]$ cd
[liveuser@localhost-live ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  utp  Videos
[liveuser@localhost-live ~]$ cd home/liveuser
bash: cd: home/liveuser: No such file or directory
[liveuser@localhost-live ~]$ cd /home/liveuser
[liveuser@localhost-live ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  utp  Videos
[liveuser@localhost-live ~]$ cd ..
[liveuser@localhost-live home]$
```

```
[liveuser@localhost-live practicas]$ cat file1
Archivo 1
1
2
3
4
5
6
final
^X
```

Nota especial:

Es preferible que los nombres de archivos y directorios en Linux no tengan caracteres especiales ni espacios en blanco. Sin embargo, si se da el caso, para pasarse a un directorio que contenga un espacio en blanco en el nombre habría que omitir dicho espacio utilizando la barra inversa. Así, si quiero entrar a un directorio llamado “aris castillo,” el comando cd debería ejecutarse de la siguiente manera:

```
cd aris\ castillo
```

¿Cómo puedo ver el contenido de un archivo en pantalla?

Para ver el contenido de un archivo se puede utilizar el comando **CAT (Concatenar)**, así **cat file1**. El resultado es que muestra en pantalla el contenido de file1.

También puede hacer esto con el comando **less**, así: **less file1**. La ventaja del comando **less** es que va mostrando pantalla por pantalla. El usuario presiona la barra espaciadora para ir avanzando y **q** para terminar. Este comando es útil cuando el archivo es muy largo.

Si lo que se desea es ver sólo el principio del archivo o el final, se pueden utilizar los comandos **head** o **tail**, así: **head x file1**; donde x es el número de líneas que desea ver, sino se especifica el comando le mostrará las 10 primeras líneas del archivo. Con el comando **tail**, sería: **tail x file1**. En este caso, x serían las últimas líneas del archivo. Si no se especifica, se mostrarán las 10 últimas líneas.

¿Se puede ver el contenido de varios archivos a la vez?

Se utiliza el comando **cat** colocando los nombres de los archivos uno después del otro, así: **cat file1 file2 file3**.

Resultado: Muestra en pantalla el contenido de cada uno de los archivos listados uno después del otro.

Resultado: Concatena archivos o entrada estándar a la salida estándar. Los resultados son diversos dependiendo de la opción utilizada. Veamos

¿Cómo creo un archivo y le agrego datos?

Para esto se puede utilizar el comando **cat** con direccionamiento desde la entrada estándar. Sintaxis: **cat > file1**.

Resultado: Crea el archivo **file1** y le permite agregar texto por el teclado. Para culminar la escritura pulse Ctrl+x. La entrada estándar se refiere a.

Si ya he cerrado el archivo, ¿cómo le agrego más datos?

Para esto también se puede utilizar el comando **cat**, utilizando doble direccionamiento. Sintaxis: **cat >> file1**.

Resultado: Permite agregar líneas de texto a un archivo existente sin modificar su contenido.

¿Cómo buscamos archivos cuando sólo tenemos cierta información de los mismos?

El comando **find** (encontrar) es sumamente útil para buscar de distintas formas archivos cuando tenemos sólo algunas pistas para proporcionar al sistema operativo.

Sintaxis: **find opción**

Resultado: Busca archivos dentro de una jerarquía de directorios.

Ejemplos:

| Ejemplo | Resultado |
|-----------------------|---|
| find / -name windows | Busca el archivo “windows” desde el directorio raíz “/” La opción name hace que el nombre sea sensitivo a las mayúsculas. Si no está seguro, puede usar la opción iname. |
| find /home -user aris | Encuentra cualquier archivo debajo del directorio home que pertenezca al usuario (opción -user) aris. |
| find /usr -name *lab* | Encuentra cualquier archivo debajo del directorio usr que tenga lab como parte del nombre (opción -name). |
| find /home mtime +120 | Encuentra cualquier archivo debajo del directorio home que tenga 120 o más días de modificación (opción -mtime). |

** Más ejemplos en:

http://linux.about.com/od/commands/a/blcmdl1_findx.htm

2. Haga sus propias pruebas para aplicar el comando. Escriba 5 ejemplos completos que funcionen de acuerdo con la configuración del equipo. En cada caso explique qué deseaba lograr y cada parámetro utilizado.

| Ejemplo | Resultado |
|--|---|
| Se busca la carpeta por el nombre que des. | <pre>[liveuser@localhost-live ~]\$ find -name laboratorio2 ./utp/laboratorio2</pre> |
| Se busca cualquier carpeta sin distinguir | <pre>[liveuser@localhost-live ~]\$ find -iname tips ./utp/laboratorio2/tips</pre> |

| | |
|---|---|
| mayúscula s o minúsculas . Es útil cuando no recordamo s. | |
| Busca las carpetas vacías | <pre>[liveuser@localhost-live ~]\$ find -empty ./.cache/tracker3/files/errors ./.cache/tracker3/files/.meta.isrunning ./.cache/abrt/applet_dirlist ./.cache/folks/avatars ./.cache/gegl-0.4/swap ./.cache/ibus ./.cache/babl ./.cache/evolution/calendar/trash ./.cache/evolution/tasks/trash ./.cache/evolution/mail/trash ./.cache/evolution/sources/trash ./.cache/evolution/addressbook/trash ./.cache/evolution/memos/trash ./Music ./Documents ./Videos ./Pictures</pre> |
| Compara la antigüeda d de 2 carpetas | <pre>[liveuser@localhost-live ~]\$ find -newer utp ./.cache/mesa_shader_cache ./.cache/mesa_shader_cache/2a ./.cache/mesa_shader_cache/2a/24147ee22e90d6754595e871b495d1dcf3e797 ./.cache/mesa_shader_cache/index ./.cache/mesa_shader_cache/bb ./.cache/mesa_shader_cache/bb/a6504f874980bd0ebde62d2a37af836ce5bf94 ./.cache/mesa_shader_cache/20 ./.cache/mesa_shader_cache/20/61f2f634a8d77f07e745ba22af169efe40997e ./utp/laboratorio2 ./utp/laboratorio2/practicas ./utp/laboratorio2/tips ./utp/laboratorio2/tips/cocina ./local/state/wireplumber ./local/state/wireplumber/restore-stream</pre> |

| | |
|--|---|
| Busca los archivos por directorio | <pre>[liveuser@localhost-live ~]\$ find -type d . ./.cache ./.cache/tracker3 ./.cache/tracker3/files ./.cache/tracker3/files/errors ./.cache/abrt ./.cache/folks ./.cache/folks/avatars ./.cache/gegl-0.4 ./.cache/gegl-0.4/swap ./.cache/gstreamer-1.0 ./.cache/ibus ./.cache/babl</pre> |
|--|---|

¿Cómo busco, no archivos, sino información dentro de los archivos?

Para esto también se utiliza el comando **less**. En este caso, se aplica primero el comando less y luego de estar en pantalla, se presiona / (slash) y la palabra a buscar. Ejemplo: less practica. Con el archivo abierto, presione /comando. Presione n para continuar.

Nota: Puede limpiar la pantalla con el comando **clear** (limpiar).

También puede buscar dentro de un archivo por palabras específicas o patrones usando el comando **grep**. Sintaxis: **grep option pattern file1**.

Resultado: Imprime en pantalla las líneas de file1 que cumplan con las opciones o patrón indicado.

Ejemplo: grep comando practica. En este caso el resultado es todas y cada una de las líneas del archivo practica en que aparezca la palabra comando.

El comando grep es **sensitivo a las mayúsculas**, así que para ignorar la mayúscula o minúscula, se debe usar la opción -i antes de la palabra a buscar, así:

grep -i comando practica

Si deseamos buscar una frase, en lugar de una sola palabra, enciérrela en apóstrofe simple, así:

```
grep -i 'spanning tree protocol' practica
```

Otras opciones del comando son:

| | |
|----------|--|
| v | para mostrar todas las líneas que no coinciden |
| n | coloca antes de cada línea el número de línea |
| c | presenta sólo el número total de líneas |

¿Cómo determino el tipo de archivo?

A través de un **ls (list)** se podría determinar el tipo de archivo viendo la extensión de los mismos. Sin embargo, hay casos en que no hay una extensión, entonces se puede utilizar el comando **file**.

Así:

file nombre-archivo. Donde **nombre-archivo** es el archivo al cual le desea aplicar el comando para conocer su tipo.

El **formato de un archivo** depende de los programas que lo usan. Puesto que los tipos de archivos no son determinados por el sistema de archivos, el núcleo no puede decir cuál es el tipo, pues no lo conoce.

El **comando file** no atiende el nombre de los archivos, ya que variedad de convenciones de nombres. Los archivos con sufijo **.c** son programas fuentes en C, pero nada impide que su contenido sea arbitrario. Lo que hace el comando **file** es leer unos cuantos bytes al principio del archivo y busca los indicios que indiquen el tipo del archivo [1].

Por ejemplo, si usted ha creado un archivo con extensión txt probablemente el resultado de file le indicará que se trata de un archivo UTF-8 Unicode Text. Esto significa que se trata de una codificación de caracteres de 8 bits del set de caracteres Unicode, compatible con ASCII. Este sistema de codificación es el más generalizado en sistemas operativos, lenguajes de programación, APIs y aplicaciones de software [2].

3. Haga la prueba con algunos archivos. Cree archivos con el comando cat. Revise con el comando file. Describa las líneas completas ejecutadas. ¿Qué resultados obtuvo? Busque el significado de los tipos de archivos de los resultados obtenidos.

```
[liveuser@localhost-live practicas]$ cat file1
Archivo 1
1
2
3
4
5
6
final
^X
[liveuser@localhost-live practicas]$ cat>file1
archivo creado con cat

^C

^X
[liveuser@localhost-live practicas]$ cat>file2
asdasd1122
salir
```

```
[liveuser@localhost-live practicas]$ file file1
file1: data
[liveuser@localhost-live practicas]$ file file2
file2: ASCII text
[liveuser@localhost-live practicas]$ file file3
file3: ASCII text
[liveuser@localhost-live practicas]$
```

¿Cómo hago copias de seguridad de mis archivos o los copio en un medio de almacenamiento externo?

El comando **cpio** permite copiar archivos en algún medio indicado por el usuario. Así:

cpio opciones <nombrearchivo> rutademedio. Donde las opciones pueden conocerse en el manual, **nombrearchivo** es el archivo que se desea copiar y **rutademedio** es la ruta donde para llegar al medio de almacenamiento deseado. Veamos un ejemplo copiando el archivo practica, creado anteriormente, a una memoria USB:

```
cpio -ov <practica> /media/30AD-6631/practica
```

En este caso, /media/30AD-6631 es la ruta creada por el sistema operativo para mi dispositivo USB. La opción -v permite desplegar el nombre de cada archivo copiado.

**** Más ejemplos en:**

<http://www.mppmu.mpg.de/Introduction/general/subsubsection2.5.8.2.2.html>

4. Haga sus propias pruebas. Describa completamente cada línea y opción utilizada.

No tenía ningún usb disponible así que no pude hacer la prueba

Retroinformación.

1. Entregue cada una de las preguntas de ejercicio.
2. Busque 5 comandos relacionados con los discutidos en esta guía. Pruébelos. Describa sus usos y escriba ejemplos específicos completos, incluyendo la sintaxis y opciones utilizadas.

| | |
|---|--|
| - | Para buscar en el directorio actual, utilizamos el elemento "." como ruta del directorio: find . <search_parameter> |
| - | Buscar todos los archivos con la extensión jpeg/JPEG o jpg/JPG: |

| | |
|---|--|
| | <code>find . -type f -iname "*.jpeg" -or -iname "*.jpg"</code> |
| - | El siguiente comando proporciona archivos de menos de 500 megabytes: <code>find . -size -500M</code> |
| - | Para encontrar los archivos que se modificaron hace apenas un día, utilizamos el parámetro de búsqueda "-mtime" seguido del valor "1": <code>find . -type f -mtime 1</code> |
| - | Para encontrar archivos totalmente accesibles para cualquier usuario, utilizamos el parámetro de búsqueda "-perm" seguido del valor "777": <code>find . -perm 777</code> |

3. ¿En qué situaciones específicas considera que serían útiles los comandos utilizados?

Creo que es importante para poder moverse con facilidad entre carpetas como lo es el "cd .." o "cd /"

Y para buscar archivos en específicos con el comando find, también el poder escribir en los archivos con cat non.

4. ¿Qué dificultades encontró durante el desarrollo del laboratorio?

Fue la primera vez que utilizaba estos comandos así que la dificultad no fue mas que leer la guía y buscar algunas cosas en internet para aprender.

5. ¿Qué mejoraría de esta experiencia de laboratorio?

El poner ejemplos con imágenes, creo que ayudaría mucho para tener mas información.

Referencias:

1. Kernighan, B. y Pike, R. El Entorno de programación Unix. Prentice Hall.

2. UTF-8. Disponible en <http://en.wikipedia.org/wiki/UTF-8>

Begginer Linux Tutorial: <http://beginnerlinuxtutorial.com/help-tutorial/basic-linux-commands>