

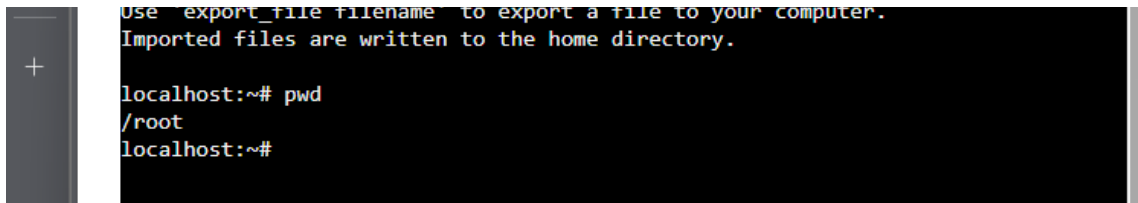
Nombre: Fecha: Grupo: 1 ☐ 2 ☐ 3 ☐ 4 ☐

PRÁCTICA 16 SISTEMAS DE DIRECTORIOS EN LINUX Y WINDOWS (II)

En la primera práctica sobre sistemas de directorios en Linux y Windows aprendimos la estructura propia del sistema de directorios de cada uno de los sistemas operativos, cómo desplazarnos por ellos, cuáles son algunos de los directorios más importantes de dichos sistemas, cómo montar unidades...

En esta segunda práctica sobre el mismo tema vamos a ampliar nuestro conocimiento sobre el sistema de directorios y vamos a aprender a crear y borrar carpetas y ficheros, crear enlaces simbólicos y enlaces “duros” a ficheros y directorios, empaquetar y comprimir carpetas, así como el modelo de gestión de ficheros propios de Linux, conocido como “inodo”.

1. Abre una terminal (un “bash”) en Ubuntu y comprueba en qué directorio te encuentras (pwd). Apúntalo en tu informe de prácticas.



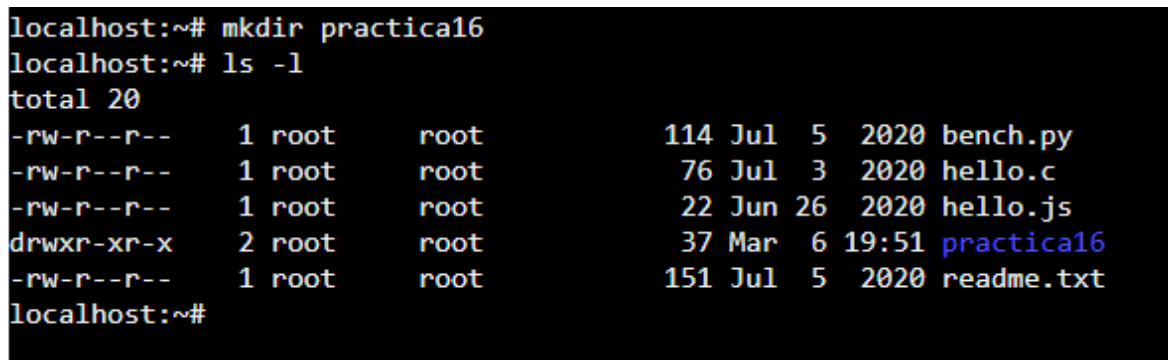
```
Use "export_file filename" to export a file to your computer.
Imported files are written to the home directory.

localhost:~# pwd
/root
localhost:~#
```

2. Crea un subdirectorio llamado practica16 (el mandato necesario es mkdir; si no conoces su sintaxis puedes usar “man mkdir”).

(en la captura 2)

3. Anota en tu informe de prácticas el propietario, permisos, y el número de directorios que contiene la carpeta creada (“ls -l”).



```
localhost:~# mkdir practica16
localhost:~# ls -l
total 20
-rw-r--r--  1 root    root      114 Jul  5  2020 bench.py
-rw-r--r--  1 root    root       76 Jul  3  2020 hello.c
-rw-r--r--  1 root    root      22 Jun 26  2020 hello.js
drwxr-xr-x  2 root    root      37 Mar  6 19:51 practica16
-rw-r--r--  1 root    root     151 Jul  5  2020 readme.txt
localhost:~#
```

El directorio que he creado tiene todos los permisos para el autor/propietario (root), lectura y escritura para el grupo y solo lectura para otros.

4. Muévete ahora al directorio creado practica16 (por medio de cd).

(en la captura 3)

5. Crea en el directorio practica16 tres subdirectorios llamados dir.uno, dir2 y dir_3 (puedes crear los tres directorios en un solo mandato, comprueba la sintaxis de mkdir en su manual) ¿Has tenido algún problema por usar los caracteres "." o "_" en los nombres de carpetas?

```
localhost:~# cd practica16/
localhost:~/practica16# mkdir dir.uno dir2 dir_3
localhost:~/practica16# ls
dir.uno  dir2    dir_3
localhost:~/practica16#
```

6. Borra el directorio dir_3 (usa el mandato rmdir). Comprueba con ls que la operación se ha completado con éxito. rmdir nos permite borrar directorios vacíos (compruébalo en el manual de rmdir). Por medio de "rm -r" podemos borrar un directorio y sus contenidos de forma recursiva (compruébalo en el manual de rm).

```
localhost:~/practica16# rmdir dir_3
localhost:~/practica16# ls -l
total 8
drwxr-xr-x  2 root    root          37 Mar  6 19:56 dir.uno
drwxr-xr-x  2 root    root          37 Mar  6 19:56 dir2
localhost:~/practica16#
```

7. Desplázate a dir.uno. Vamos a crear ahora un fichero de texto por medio del editor nano. Ejecuta nano en la terminal. Ahora debes escribir un fichero de texto con los siguientes datos:

Nombre
Apellidos Fecha
Dedicación

Guárdalo con el nombre datos_personales. Comprueba por medio de "ls" que está bien creado.

```
localhost:~/practica16/dir.uno# ls -l
total 4
-rw-r--r--  1 root    root          37 Mar  6 20:02 datos_personales
localhost:~/practica16/dir.uno#
```

8. Desplázate al directorio dir2 y crea dos ficheros de texto por medio de nano, de nombre prueba1 y prueba2. El contenido de los mismos queda a tu elección (pero no los dejes vacíos; conviene que sean distintos).

```
GNU nano 4.9.3 prueba1
GNU nano 4.9.3 prueba2
```

```
localhost:~/practica16/dir2# ls
prueba1 prueba2
localhost:~/practica16/dir2#
```

9. Sin salir de la carpeta dir2, copia el fichero datos_personales (que está en dir.uno) en otro llamado datos_personales.copia en dir2. El mandato que permite copiar ficheros (o directorios) es cp. Recuerda que para “subir un nivel” en el árbol de directorios puedes utilizar la abreviatura “..”. Comprueba que la operación se ha completado con éxito por medio de ls.

```
localhost:~/practica16/dir2# ls
prueba1 prueba2
localhost:~/practica16/dir2# cp ../dir.uno/datos_personales datos_personales.copia
localhost:~/practica16/dir2# ls
datos_personales.copia prueba1 prueba2
localhost:~/practica16/dir2# more datos_personales.copia
Nombre
Apellidos Fecha
Dedicacion

localhost:~/practica16/dir2#
```

10. En la misma carpeta dir2 vamos a hacer una copia (oculta) del fichero datos_personales.copia en un fichero .datos_personales. Comprueba que el archivo está en la carpeta por medio de ls -a. Comprueba por medio de ls que el archivo no es visible.

```
localhost:~/practica16/dir2# ls -l
total 12
-rw-r--r-- 1 root root 37 Mar 6 20:07 datos_personales.copia
-rw-r--r-- 1 root root 19 Mar 6 20:04 prueba1
-rw-r--r-- 1 root root 19 Mar 6 20:04 prueba2
localhost:~/practica16/dir2# ls -a
.
..
.datos_personales. prueba1
datos_personales.copia prueba2
localhost:~/practica16/dir2#
```

11. El mandato `cp` también nos permite copiar directorios (con sus contenidos). Generalmente, si queremos copiar directorios queremos que se copien también todos sus ficheros y subdirectorios. Para ello debemos usar la opción “-r” (copia recursiva). Ejecuta el siguiente mandato

`$cp -r /usr/games .`

```
localhost:~# cd ..
localhost:/# ls
bin    etc    lib    mnt    proc   run    srv    tmp    var
dev    home  media  opt    root   sbin   sys    usr
localhost:/# cd tmp/
localhost:/tmp# mkdir games
localhost:/tmp# cd games/
localhost:/tmp/games# echo "esto es info de games" > gamesinfo.bin
localhost:/tmp/games# ls
gamesinfo.bin
localhost:/tmp/games# cd ../../
localhost:/# ls
bin    etc    lib    mnt    proc   run    srv    tmp    var
dev    home  media  opt    root   sbin   sys    usr
localhost:/# cd usr
localhost:/usr# cp -r /tmp/games .
localhost:/usr# ls
bin            include        local
games          lib            sbin
i586-alpine-linux-musl  libexec       share
localhost:/usr# cd games
localhost:/usr/games# more games
more: stat of games failed: No such file or directory
localhost:/usr/games# ls
gamesinfo.bin
localhost:/usr/games# more games*
esto es info de games
localhost:/usr/games#
```

¿A qué directorio has copiado la carpeta “games”? Ejecuta el mandato “`ls -l`”. ¿Qué directorio abreviamos por “.”? Con la orden anterior has realizado una copia completa de la carpeta “games”.

Dirígete a la carpeta “games” del directorio “dir2”. Ejecuta el mandato “`ls -l`”. Comprueba si los ficheros disponen de permiso de ejecución (“x”).

¿Sabrías ejecutar alguno de los ficheros (ejecutables) de la carpeta? (Recuerda cómo ejecutábamos el programa "GoogleEarth.bin" en la práctica de instalación de paquetes). Trata de ejecutar alguno de los ficheros de la carpeta.

¿Qué sucederá si eliminamos alguno de los ficheros que se encuentran en la nueva carpeta creada "games"? ¿Desaparecerá también de la carpeta del sistema "/usr/games"?

Elimina uno cualquiera de los ficheros por medio del mandato rm. Apunta su nombre. Comprueba con ls que la operación se ha completado con éxito. Lista ahora el contenido de la carpeta "/usr/games" (no hace falta que te desplaces hasta ella, "ls -l /usr/games" lo hará directamente). ¿Aparece el fichero que has eliminado en tu copia de la carpeta "games"?

Una vez hemos copiado los ficheros, cada uno tiene su propia entidad, y eliminar o modificar uno de ellos (el original o la copia) no tiene consecuencias sobre el otro. Los dos ficheros son entidades distintas.

12. Además de copiar ficheros y directorios, también podemos mover ficheros entre directorios. Vamos a mover el fichero "datos_personales" de la carpeta "dir.uno" a "dir2". Para ello puedes hacer uso del mandato "mv". La sintaxis es "mv fichero destino" (recuerda que puedes usar ".." para subir en el árbol de directorios).

2 - 7

13. Intenta mover el fichero que has borrado antes en tu copia de la carpeta "games" en dir2 desde la carpeta /usr/games hasta tu copia de la carpeta games.

¿Qué respuesta has obtenido? ¿Cómo puedes evitar la limitación encontrada? Compruébalo.

Apunta en tu informe la diferencia entre "cp" y "mv". Por medio de cp hemos creado una copia de un fichero existente (y el sistema no nos advirtió de ningún error), mientras que por medio de "mv" hemos cambiado el fichero de lugar.

14. El mandato mv también nos permite mover carpetas. Mueve el directorio dir.uno al escritorio. Comprueba que la operación se ha completado con éxito.

15. Por medio de mv también podemos renombrar carpetas y ficheros. Renombra el directorio dir.uno como mis_textos. Comprueba el resultado de la operación (ha debido desaparecer dir.uno y aparecer una nueva carpeta mis_textos).

16. Pasamos ahora a ver cómo se pueden crear enlaces a ficheros. Sitúate en el escritorio y crea un enlace débil o simbólico a uno de los ficheros que contiene tu carpeta games (que debería estar en /home/alumno/practica16/dir2/games). Para ello debes usar el mandato:

```
$ln -s ../practica16/dir2/games/nombre_del_juego enl_debil_juego
```

Comprueba ahora con "ls -l" las características del enlace creado. ¿Qué tipo de elemento es? (Recuerda que eso lo podemos saber por la primera letra de su formato largo: "d" era válido para directorio, "-" para fichero, "l" para enlaces). ¿Qué información adicional contiene la línea sobre el enlace? ¿Puedes saber a qué directorio se refiere?

17. ¿Para qué sirve el enlace simple? Trata de ejecutar ahora el fichero creado enl_debil_juego. ¿Qué ha sucedido? ¿Qué programa se ejecuta? Observa el icono que el gestor de ventanas de Ubuntu le ha asignado al enlace débil. A diferencia de la copia que realizamos antes, el enlace débil sólo es una redirección al fichero original; si modificamos el fichero original, el enlace débil, que accederá a ese mismo fichero, también "verá" las modificaciones. Si eliminamos el fichero original, el enlace dejará de ser útil (apuntará a un fichero que no existe).
18. Veamos ahora los enlaces fuertes. En el mismo escritorio, ejecuta el siguiente mandato (observa la diferencia con el mandato anterior; utiliza el mismo nombre_del_juego que antes): `$ln ../practica16/dir2/games/nombre_del_juego enl_fuerte_juego` Trata de ejecutar el fichero enl_fuerte_juego. ¿Qué sucede?

3 - 7

19. Aunque pueda parecer lo contrario por su comportamiento, existen diferencias importantes entre enl_debil_juego y enl_fuerte_juego. Ejecuta el mandato "ls -l". Observa las diferencias entre el enlace débil y el fuerte. ¿Qué tamaño tiene cada uno de los ficheros? ¿Existen diferencias entre los permisos? ¿Aparece en enl_fuerte_juego la referencia a algún otro directorio?
20. Ejecuta ahora el mandato "ls -li". Verás que en la columna izquierda de la respuesta obtenida ha aparecido un número entero. Apunta el número de los archivos "enl_fuerte_juego" y "enl_debil_juego". Este número es lo que se conoce en los sistemas de ficheros propios de Unix como "inodo". Puedes encontrar información sobre los mismos en <http://es.wikipedia.org/wiki/Inodo>. Apunta las principales características de los inodos.

Comprueba el inodo correspondiente al fichero de la carpeta games (en dir2) que has enlazado. ¿Coincide con alguno de los inodos de los enlaces simbólico o fuerte? ¿Coincide su tamaño con el de alguno de los enlaces de que disponías?

21. Ahora, en tu carpeta games, borra el fichero al que has creado los enlaces (rm nombre_del_juego). Vuelve al escritorio. Vuelve a ejecutar "ls -l". Trata de ejecutar enl_fuerte_juego y apunta el resultado. Trata de ejecutar enl_debil_juego y apunta el resultado.

El enlace fuerte apunta a la dirección de memoria en que se encuentra el fichero original, mientras que el enlace débil apuntaba al fichero original. De ese modo, cuando borramos el fichero original, el enlace fuerte sigue funcionando, mientras que el enlace débil deja de hacerlo.

22. A partir de la información que has leído en el enlace anterior y de tus propias conclusiones, trata de explicar la diferencia entre los enlaces simbólicos, enlaces fuertes, copias de ficheros (cp) y desplazamientos de los mismos (mv). Debe quedar claro que cada una de las 4 posibilidades anteriores es distinta, y en qué sentido lo son.

23. También podemos crear enlaces fuertes o simbólicos entre carpetas. Vamos a crear un enlace simbólico al directorio propio del usuario raíz:

```
$ln -s /root raiz
```

Muévete a la carpeta “raiz”. Comprueba su contenido con “ls -lia”. Comprueba ahora los contenidos de la carpeta “/root” por medio de:

```
$ls -lia /root
```

¿Obtienes los mismos contenidos? A partir de ahora, la carpeta “raiz” de nuestro escritorio nos ofrece un enlace a la carpeta “/root”. Si quieres eliminar el enlace, puedes hacerlo por medio de “rm raiz” (eliminando sólo el enlace, no los contenidos originales).

24. Por último, vamos a ver algunos mandatos de empaquetado y desempaquetado propios de Linux. Un programa muy utilizado para

4 - 7

empaquetar archivos y carpetas es tar. Puedes encontrar información sobre el mismo en <http://es.wikipedia.org/wiki/Tar>. En realidad tar, por defecto, no comprime información, simplemente la empaqueta o junta en un solo fichero. Ejecuta en la shell el mandato “man tar”.

Vamos a empaquetar ahora los contenidos de la carpeta practica16. Dirígete a tu directorio \$HOME. Ejecuta el mandato:

```
$tar -cvf practica16.tar practica16
```

Apunta lo que hace cada una de las opciones “c”, “v”, “f”. Ejecuta ahora “ls -l” y comprueba el nuevo archivo que ha aparecido. Los ficheros producidos por tar tienen algunas características importantes, como que mantienen los permisos de los archivos originales, preservan los enlaces simbólicos o fuertes, mantienen el árbol de directorios... Por eso constituye una herramienta ideal para trasladar información entre máquinas Linux.

25. Para poder descomprimir un archivo tar hay que realizar una acción parecida. Borraremos en primer lugar la carpeta practica16. Para ello debes ejecutar “rm -r practica16”. Comprueba que has obtenido el resultado deseado (la carpeta debe haber desaparecido). Ahora vamos a desempaquetar el archivo practica16.tar:

```
$tar -xvf practica16.tar
```

Apunta el significado de la opción “x”. Comprueba el resultado por medio de “ls -l”.

26. Además de utilidades para empaquetar, pero que no reducen ni disminuyen el tamaño de los archivos originales, el intérprete de mandatos también nos presta ciertas aplicaciones que permiten comprimir carpetas o ficheros. Dos de las más conocidas son bzip2 y gzip. bzip2 consigue mayores niveles de compresión, pero también requiere un mayor uso de CPU. El uso de ambas es similar. Apunta el tamaño del fichero practica16.tar (ejecuta “ls -l” y elige la columna correspondiente).

Ejecuta ahora el mandato:

```
$gzip practica16.tar
```

Comprueba el tamaño del archivo obtenido (“ls -l”).

Descomprímelo por medio de:

```
$gunzip practica16.tar.gz
```

Vuelve a comprimirlo por medio de:

```
$gzip -9 practica16.tar
```

La opción “9” le indica al programa que trate de conseguir el máximo nivel de compresión. Comprueba su tamaño. Descomprímelo.

5 - 7

El funcionamiento de bzip2 es similar:

```
$bzip2 practica16.tar
```

Comprueba su tamaño.

```
$bunzip2 practica16.tar.bz2
```

El uso combinado de tar y bzip2 o gzip también es posible. En general es bastante común que encuentres ficheros para programas o aplicaciones que tengan extensión “.tar.gz” o “.tar.bz2”, y que deberías ser capaz de descomprimir con los mandatos anteriores. Puedes encontrar las opciones combinadas correspondientes en <http://es.wikipedia.org/wiki/Tar>.

27. Vuelca todos los mandatos de la sesión por medio de history a un fichero “mandatos_practica_16”.

Vamos a realizar alguno de los ejercicios anteriores en Windows.

28. Abre una consola cmd en Windows.

29. Crea un subdirectorio llamado practica16 por medio de mkdir.

30. Muévete al directorio creado practica16 (por medio de cd).

31. Crea en el directorio practica16 tres subdirectorios llamados dir.uno, dir2 y dir_3. ¿Has tenido algún problema por usar los caracteres “.” o “_” en los nombres de carpetas?

32. Borra el directorio dir_3 (usa el mandato rmdir o rd). Comprueba con dir que la operación se ha completado con éxito. A diferencia del comando gnu de Linux, rmdir nos permite borrar directorios vacíos o con contenidos. Por medio de “rmdir /S” podemos borrar un directorio y sus contenidos de forma recursiva.

33. Desplázate a dir.uno. Vamos a crear ahora un fichero de texto por medio del editor edit. Ejecuta edit en la terminal. Ahora debes escribir un fichero de texto con los siguientes datos:

Nombre

Apellidos Fecha Dedicación

Guárdalo con el nombre datospersonales.

34. Desplázate al directorio dir2 y crea dos ficheros de texto por medio de edit, de nombre prueba1 y prueba2. El contenido de los mismos queda a tu elección.

35. Borra el fichero prueba2. El mandato para borrar ficheros en cmd es “del”.

6 - 7

36. Sin salir de la carpeta dir2, copia el fichero datospersonales (que está en dir.uno) en otro llamado datospersonales.copia en dir2. El mandato que permite copiar ficheros (o directorios) es copy. Recuerda que para “subir un nivel” en el árbol de directorios puedes utilizar la abreviatura “..”, y que en Windows la barra de las rutas es “\”. Comprueba que la operación se ha completado con éxito por medio de dir.

El equivalente al mandato “cp -r” en GNU lo podemos encontrar en el mandato xcopy de Windows, que nos permite copiar directorios (con sus contenidos).

37. Además de copiar ficheros y directorios, también podemos mover ficheros entre directorios. Vamos a mover el fichero “datospersonales” de la carpeta “dir.uno” a “dir2”. Para ello puedes hacer uso del mandato “move”. La sintaxis es “move fichero destino” (recuerda que puedes usar “..” para subir en el árbol de directorios).

38. El mandato move también nos permite mover carpetas. Mueve el directorio dir.uno al escritorio. Comprueba que la operación se ha completado con éxito.
39. Por medio de rename podemos renombrar carpetas y ficheros. Renombra el directorio dir.uno como mis_textos. Comprueba el resultado de la operación (ha debido desaparecer dir.uno y aparecer una nueva carpeta mis_textos).
40. En Windows también existe la posibilidad de crear enlaces a ficheros o carpetas. En realidad lo que se hace en Windows es equivalente a los “enlaces débiles” de Linux; se crean ficheros que apuntan a un fichero original. Si por algún motivo el fichero original desaparece, todos los enlaces quedan inutilizados. Puedes encontrar la sintaxis y algunos ejemplos en [http://technet.microsoft.com/en-us/library/cc753194\(WWS.10\).aspx](http://technet.microsoft.com/en-us/library/cc753194(WWS.10).aspx).
41. Utiliza el mandato DOSKEY /H > mandatos_windows_practica_16 para recuperar la historia de la sesión.
42. Sube a tu página de inicio en belenus el informe de la práctica, así como los ficheros mandatos_practica_16 y mandatos_windows_practica_16.