**Anotaciones**:

++

*-Realiza los siguientes ejercicios sobre máquinas virtuales. En lugar de realizar capturas de pantalla como en otras prácticas, para cada apartado indica el comando que has utilizado en un recuadro (tabla).*

*-En aquellos apartados (si los hubiere) donde se pregunta por el resultado de un comando indica por escrito la salida que te muestra el sistema.*

*-En el documento que entregues el enunciado de los ejercicios debe ir en un color y las respuestas en otro color.*

**Actividades**:

**1. Abre una terminal (un “bash”) en Ubuntu y comprueba en qué directorio te encuentras (pwd).**

Me encuentro en /home/usuario

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

Usamos el comando mkdir práctica

**3. Indica el propietario, permisos, y el número de directorios que contiene la carpeta creada (“ls -l”).**

El propietario es usuario. Los permisos son: drwxrwxr-x. Contiene 0 directorios

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

Para ello usamos el comando cd práctica/

**5. Crea en el directorio practica 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?**

el comando usado es: *mkdir dir.uno dir2 dir\_3*

**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).**

* Para borrar usamos el comando: rmdir dir\_3
* Y con ls comprobamos que ya no aparece el directorio llamado dir\_3

**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.**

Usamos el comando *nano datos.personales*. Una vez dentro de nano escribimos los datos necesario y guardamos el fichero

**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).**

Solo usaremos el comando cd ../dir2 para dirigirnos hacia el directorio y dentro de él, con el comando nano prueba1 y nano prueba2 crearemos y guardaremos los ficheros creados

**9. Sin salir de la carpeta dir2, copiar 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.**

Comando: cp ../dir.uno/datos.personales datos.personales\_copia

**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. Los archivos ocultos en los sistemas Linux se caracterizan únicamente por que su nombre empieza por “.”.**

**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**

**sudo cp –r /var/log .**

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

Hemos copiado log a la carpeta en la que nos encontramos (dir2)

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

Solo algunos ficheros contienen el permiso de ejecución

**¿Qué sucederá si eliminamos alguno de los ficheros que se encuentran en la nueva carpeta creada “log”? ¿Desaparecerá también de la carpeta del sistema “/var/log”?**

Solo se eliminaran. No, ya que este directorio de log es una copia de el original

**Elimina uno cualquiera de los ficheros por medio del mandato rm. Apunta su nombre.**

sudo rm -r -f btmp

Con estas opciones podremos eliminar sin pedir confirmación y de manera directa

**Comprueba con ls que la operación se ha completado con éxito. Lista ahora el contenido de la carpeta “/var/log” (no hace falta que te desplaces hasta ella, “ls –l /var/log” lo hará directamente). ¿Aparece el fichero que has eliminado en tu copia de la carpeta “log”?**

Podremos comprobar como en la carpeta sigue apareciendo el archivo borrado btmp

**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).**

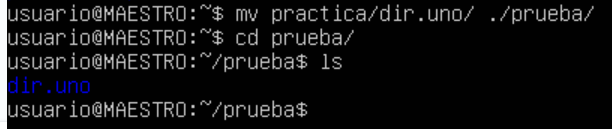
Comando: mv ../dir.uno/datos.personales dir2

**13. Intenta mover el fichero que has borrado antes en tu copia de la carpeta “log” en dir2 desde la carpeta /var/log hasta tu copia de la carpeta log.**

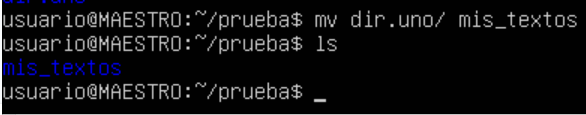
**¿Qué respuesta has obtenido? ¿Cómo puedes evitar la limitación encontrada? Compruébalo.**

**Apunta en tu documento 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 a una carpeta prueba que deberás crear previamente en tu Home Comprueba que la operación se ha realizado correctamente.**



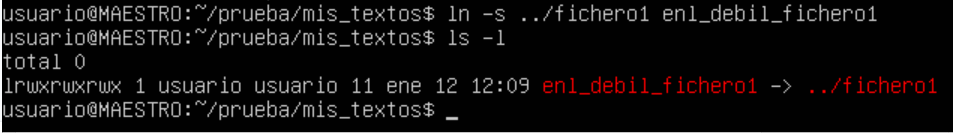
**15. Por medio de mv también podemos renombrar carpetas y ficheros. Renombra el directorio dir.uno como mis\_textos. Comprueba el resultado de la operación (ha debido desaparecer dir.uno y aparecer una nueva carpeta mis\_textos).**



**16. Pasamos ahora a ver cómo se pueden crear enlaces a ficheros. Sitúate en tu directorio home y crea un fichero llamado fichero1 con nano. Introduce algún texto. A continuación entra en la subcarpeta mis\_textos y crear ahí un enlace débil a fichero1:**

*$ln –s ../fichero1 enl\_debil\_fichero1*

**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?**



Es un elemento Enlace, esto lo sabemos porque su primera letra es L. Se refiere al directorio donde se encuentra el fichero1

**17. ¿Para qué sirve el enlace débil? Trata de abrir ahora el fichero creado enl\_debil\_fichero (puedes usar nano).**

**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). Realiza la prueba de modificar el fichero y abrir el enlace.**

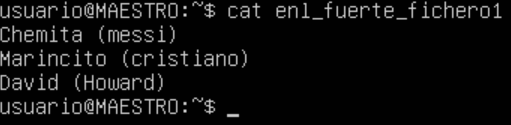


Este es el fichero enlace, obviamente vacío, pero el cual podemos modificar

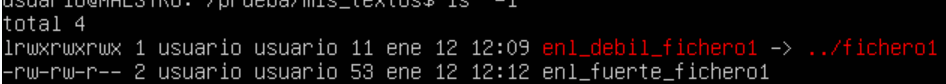
**18. Veamos ahora los enlaces fuertes. En el mismo directorio, ejecuta el siguiente mandato (observa la diferencia con el mandato anterior; utiliza el mismo nombre\_del\_fichero que antes):**

*$ln ../fichero1 enl\_fuerte\_fichero1*

**Visualiza el contenido del enlace fuerte: cat enl\_fuerte\_fichero1**



**19. Aunque pueda parecer lo contrario por su comportamiento, existen diferencias importantes entre enl\_debil\_fichero1 y enl\_fuerte\_fichero1. 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\_fichero1 la referencia a algún otro directorio?**



El fichero fuerte tiene una capacidad de 53 y el débil de 11. Si el enlace fuerte tiene permisos de lectura y escritura sobre el archivo, sin embargo el débil tiene todos los permisos. No no aparece ninguna otra referencia

**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\_fichero1” y “enl\_debil\_fichero1”. 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 home 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 disponíamos?**

* Enlace Fuerte: 143
* Enlace Débil: 59

Es una estructura de datos propia de los sistemas de archivos tradicionalmente empleados en los sistemas operativos tipo Unix. Contiene las características de un archivo regular, directorio, o cualquier otro objeto que pueda contener el sistema de ficheros

No, todos los inodos tienen un número diferente

**21. Ahora, en tu carpeta home, borra el fichero al que has creado los enlaces (rm fichero1). Vuelve a la subcarpeta mis\_textos. Vuelve a ejecutar “ls - l”. Trata de abrir enl\_fuerte\_fichero1 y apunta el resultado. Trata de ejecutar enl\_debil\_fichero1 y apunta el resultado.**

El enlace fuerte podemos seguir abriendo y nos sigue apareciendo editado, sin embargo el enlace débil no podemos abrirlo

**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.**

1. **CP**:
   * Crea una copia independiente del archivo original
   * Los cambios en la copia no afectan al original
2. **Enlaces Simbólicos**

* Si el archivo original se elimina, el enlace simbólico se vuelve inútil
* Son referencias a otros archivos por su ruta

1. **Enlaces Fuertes**:

* Son referencias directas al mismo contenido en disco
* Cuando se elimina uno de los enlaces, el contenido permanece hasta que se eliminen todos los enlaces

1. **MV**:

* Mueve un archivo o directorio de una ubicación a otra en el sistema de archivos
* No crea una nueva copia, simplemente cambia la ubicación del archivo original

**23. También podemos crear enlaces fuertes o simbólicos entre carpetas. Vamos a crear un enlace simbólico al directorio /var/log:**

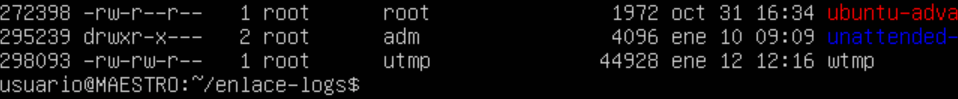
$ln –s /var/log enlace-logs

**Muévete a la carpeta “enlace-logs”. Comprueba su contenido con “ls -lia”. Comprueba ahora los contenidos de la carpeta “/var/log” por medio de:**

$ls –lia /var/log

**¿Obtienes los mismos contenidos? A partir de ahora, la carpeta “enlace-logs” de nuestro home nos ofrece un enlace a la carpeta “/var/log”. Si quieres eliminar el enlace, puedes hacerlo por medio de “rm Enlace-logs” (eliminando sólo el enlace, no los contenidos originales).**

Podemos ver el mismo contenido a través de la carpeta enlace-logs

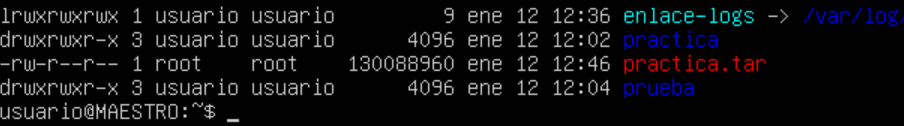


**24. Por último, vamos a ver algunos mandatos de empaquetado y desempaquetado propios de Linux. Un programa muy utilizado para empaquetar archivos y carpetas es tar. Puedes encontrar información sobre el mismo en http://es.wikipedia.org/wiki/Tar . En realidad tar, por defecto, no comprime información, 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 práctica. Dirígete a tu directorio $HOME. Ejecuta el mandato:**

*$tar –cvf practica.tar practica*

**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.**



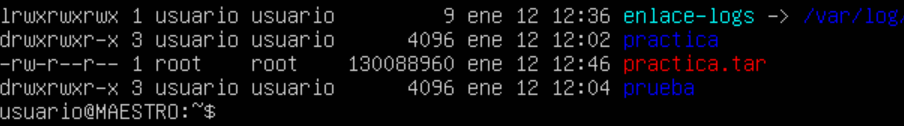
Opciones

* **c - Crear un nuevo archivo tar:**
* **v - Modo verbose:**
* **f - Especificar el nombre del archivo tar:**

**25. Para poder descomprimir un archivo tar hay que realizar una acción parecida. Borramos en primer lugar la carpeta practica. Para ello debes ejecutar “rm –r practica”; responde afirmativamente a todas las preguntas que se te planteen. Comprueba que has obtenido el resultado deseado (la carpeta debe haber desaparecido). Ahora vamos a desempaquetar el archivo practica.tar:**

*$tar –xvf practica.tar*

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



* x: Indica que se realizará la operación de extracción (descomprimir) del archivo tar.

**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 practica.tar (ejecuta “ls -l” y elige la columna correspondiente).**

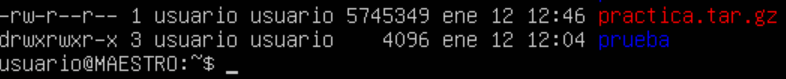
**TAMAÑO:**

* **130088960**

Ejecuta ahora el mandato:

$gzip practica.tar

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



Descomprímelo por medio de:

$gunzip practica.tar.gz

Vuelve a comprimirlo por medio de:

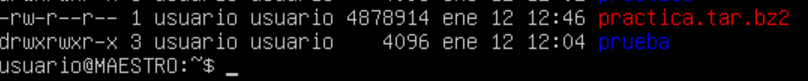
$gzip -9 practica.tar

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

El funcionamiento de bzip2 es similar:

$bzip2 practica.tar

Comprueba su tamaño.



$bunzip2 practica.tar.bz2



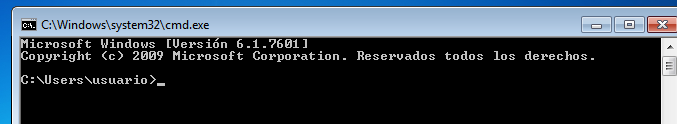
**El uso combinado de tar y bzip2 o gzip también es posible. En general es bastante común que encuentres ficheros para programas o aplicaciones que tengan extensión “.tar.gz” o “.tar.bz2”, y que deberías ser capaz de descomprimir con los mandatos anteriores. Puedes encontrar las opciones combinadas correspondientes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

**27. Vuelca todos los mandatos de la sesión por medio de history a un fichero “mandatos\_practica”. History > mandatos\_practica**

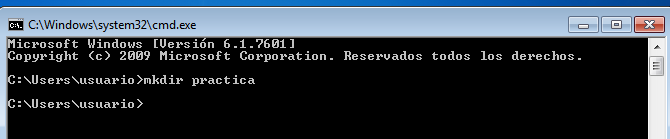


**Vamos a realizar alguno de los ejercicios anteriores en Windows.**

**28. Abre una consola cmd en Windows.**



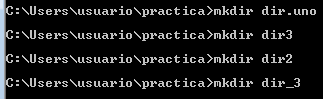
**29. Crea un subdirectorio llamado práctica por medio de mkdir.**



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



**31. Crea en el directorio practica tres subdirectorios llamados dir.uno, dir2 y dir\_3. ¿Has tenido algún problema por usar los caracteres “.” o “\_” en los nombres de carpetas?**



Para crear los subdirectorios no he tenido ningun problema, solo tenemos que saber que cuando escribes con espacio es necesario poner “” al inicio y final

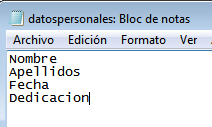
**32. Borra el directorio dir\_3 (usa el mandato rmdir o rd). Comprueba con dir que la operación se ha completado con éxito. A diferencia del comando gnu de Linux, rmdir nos permite borrar directorios vacíos o con contenidos. Por medio de “rmdir /S” podemos borrar un directorio y sus contenidos de forma 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.**

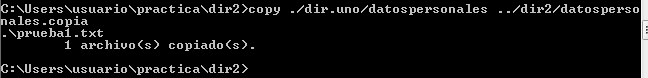


**35. Borra el fichero prueba2. El mandato para borrar ficheros en cmd es “del”.**

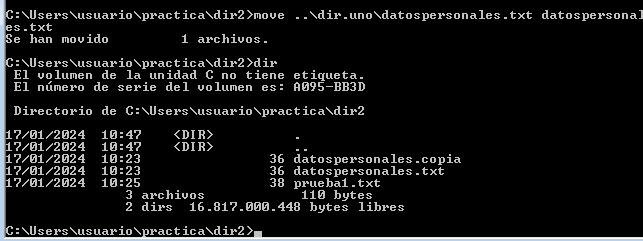


**36. Sin salir de la carpeta dir2, copiar el fichero datospersonales (que está en dir.uno) en otro llamado datospersonales.copia en dir2. El mandato que permite copiar ficheros (o directorios) es copy. Recuerda que para “subir un nivel” en el árbol de directorios puedes utilizar la abreviatura “..”, 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).**



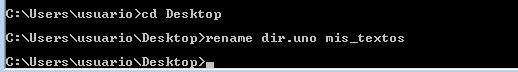
**37. Además de copiar ficheros y directorios, también podemos mover ficheros entre directorios. Vamos a mover el fichero “datospersonales” de la carpeta “dir.uno” a “dir2”. Para ello puedes hacer uso del mandato “move”. La sintaxis es “move fichero destino” (recuerda que puedes usar “..” para subir 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(WS.10).aspx.**

