REDES DE ORDENADORES

1. Objetivos

- Aprender los comandos básicos para la manipulación de ficheros y directorios dentro de un determinado Sistema Operativo, en este caso Linux (Ubuntu).

2. Introducción

Para desenvolverse en cualquier entorno es imprescindible tener unas nociones básicas de la manipulación de ficheros y directorios que el Sistema Operativo con el que estamos trabajando nos ofrece.

Dentro de un entorno como Windows, es sencillo realizar todo este tipo de tareas cotidianas, ya que, con los entornos de ventanas gráficas, las operaciones se limitan a simples acciones de ratón o teclado.

Linux (Ubuntu) también nos ofrece una interfaz gráfica que nos permite manipular todos los elementos a nuestra disposición, pero no siempre es así, con lo que resulta necesario conocer los comandos básicos para poder manipular nuestra información por medio del uso de la línea de comandos en modo texto.

3. Arranque

Vamos a arrancar nuestro ordenador y elegiremos la opción de Ubuntu dentro del menú de opciones de Sistemas Operativos al inicio (*Grub*).
El sistema Ubuntu instalado en el laboratorio arranca con el usuario ADMIN1 y la contraseña 123456.

Una vez dentro accederemos al modo gráfico, pudiendo manipular carpetas y archivos visualmente. Pero, el objetivo de la práctica es manipular archivos y carpetas en modo texto. **Alt + F2** con la palabra **gnome-terminal** (Accedemos a la shell)

4. Comienzo

El alumno realizará la práctica de individualmente y se permite consultar en Internet cualquier duda. Deberá escribir la salida de los comandos ejecutados.

Primero de todo vamos a comprobar en qué directorio estamos, para ello ejecutaremos: **pwd** (return):

A continuación, vamos a listar el contenido de nuestro directorio actual. Para ello escribimos el comando:
ls (return):

Con ello veremos la lista de directorios y ficheros que contiene el directorio actual. Vamos a crear una carpeta en el que realizaremos todas las pruebas de la práctica, por ejemplo, *practica0*. Para ello utilizamos el comando <*mkdir nombrecarpeta*>. En nuestro caso:
mkdir practica0 (return):

Si volvemos a listar el contenido del directorio **ls** (return):

Comprobaremos que la carpeta para la práctica ha sido creada.

Para entrar en este directorio teclearemos la instrucción <*cd nombre carpeta*>. En nuestro caso:
cd practica0 (return):

Comprobar también la ruta de directorios que nos especifica ahora el pwd:
pwd (return):

Para salir de la carpeta de la práctica se ejecuta el comando:
cd .. (return):

El comando *echo* muestra en la shell la cadena que se le especifica como parámetro. Ejemplo:
echo Hola a todos (return) :

O bien muestra el valor de una determinada variable del sistema si ésta va precedida por $:
echo $HOME (return) :

echo $HOSTNAME (return) :

Una vez vistos estos comandos básicos, pasamos a trabajar con ficheros.
Comprueba que estás fuera del directorio *practica0* mediante el comando **pwd** y si estás dentro, vuelve al directorio anterior mediante **cd ....**
En el directorio *home* vamos a crear un fichero donde realizaremos algunas pruebas. Para ello utilizaremos el editor de texto **vi**.
Para crear el fichero utilizaremos la instrucción <*vi nombre_fichero*>. Por ejemplo, vamos a darle el nombre prueba al fichero:
vi prueba (return) :

Con este comando entraremos en el editor vi para modificar el fichero que hemos creado. Una vez en él, vamos a poner varias líneas de comandos de la shell para ejecutarlos por medio del fichero. Al entrar en el vi, estamos en modo comando, con lo que todo lo que se teclee se intentará identificar con instrucciones del vi. Para pasar al modo inserción y poder escribir, pulsar la tecla 'i' y escribir las siguientes líneas:
#Prueba de edicion (return) echo $HOME (return) echo $HOSTNAME (return) pwd (return)

Una vez escritas estas líneas, pasaremos al modo comando del vi mediante la tecla **ESC**, y pulsaremos
:wq (return) :

Para grabar los cambios realizados y salir del editor.

Estaremos de nuevo en la línea de comandos de linux. Si listamos el

contenido del directorio de nuevo
ls (return):

Veremos que el fichero ha sido creado y está en el directorio actual.
A continuación, moveremos el fichero a la carpeta de la práctica que hemos creado. Para ello usaremos el comando
<*mv nombrejichero directorio_destino*>. En nuestro caso:
mv prueba practica0 (return) :

Si comprobamos de nuevo el contenido del directorio actual
ls (return):

Veremos que ya no existe el fichero que hemos creado anteriormente, ya que ha sido movido mediante la instrucción anterior. Comprobarlo usando la instrucción <*ls nombre_carpeta*>, que permite comprobar el contenido de un directorio concreto
ls practica0 (return) :

O bien entrando en la carpeta *practica0* y haciendo un listado normal del contenido.

cd practica0 (return):

ls (return):

Dentro de la carpeta de la práctica (si no hemos ejecutado el comando cd practica0 anterior, ejecutarlo ahora), intentamos ejecutar el fichero que hemos creado:
./prueba (return) :

Podremos comprobar que no es posible ejecutarlo, ya que el fichero no tiene el permiso de ejecución (x) necesario. Listar el contenido del directorio con la opción de ver todos los detalles de los archivos (incluido los permisos):
ls -l (return) :
Vemos que el fichero ha sido creado con los permisos de lectura(r) y escritura(w). Vamos a cambiarle los permisos para poder ejecutarlo y comprobar su funcionamiento. Para ello utilizaremos la instrucción <*chmod [opciones][permisos] nombre_fichero/directorio*>.
Los permisos que podemos darle (+) / quitarle (-) al fichero son los de lectura(r)/escritura(w)/ejecución(x). Vamos a darle el permiso de ejecución que necesita para poder ser ejecutado:
chmod +x prueba (return) :

Si comprobamos de nuevo los permisos veremos que ahora el fichero los posee todos:

ls -l (return) :

Comprobar que también se le puede quitar permisos, por ejemplo:
chmod -r prueba (return) :

O darle más de un permiso a la vez:
chmod +rw prueba (return)

Con el permiso de ejecución podemos ejecutar nuestra prueba para ver que sale.
./prueba (return):

Al ejecutar este comando el sistema ejecuta todas las instrucciones contenidas en nuestro fichero, de manera que deberíamos obtener información del directorio HOME, el nombre de la máquina y la ruta de directorios actual.

A continuación crearemos dos nuevas carpetas, carpeta1 y carpeta2, mediante el comando *mkdir*.

mkdir carpeta1 (return) :
mkdir carpeta2 (return) :

Mover la *carpeta2* dentro de la *carpeta1* mediante el comando *mv*.
mv carpeta2 carpeta1 (return) :

y copiaremos el fichero de prueba dentro de la carpeta1 mediante la instrucción
<*cp fichero origen fichero_destino*>, de la siguiente manera:
cp prueba carpeta1/prueba2 (return) :

Para copiarlo dentro de la nueva carpeta, en el *fichero_destino* indicaremos la carpeta nueva a la que va y el nombre que recibirá el fichero que se copia dentro de ella.
Si hacemos un
ls carpeta1 (return) :

Veremos que contiene tanto la *carpeta2* como el nuevo fichero copiado, *prueba2*.

Podemos también cambiar de nombre al nuevo fichero copiado. Para ello, entrar en la carpeta carpeta1 y ejecutar el comando <*mv nombre actual nuevo_nombre*> que permite cambiar el nombre del fichero que se especifica. Los pasos son los siguientes:
cd carpeta1 (return) :

mv prueba2 pruebaFin (return) :

ls -l (return) :

Por último vamos a borrarlo todo. Para ello se utiliza la instrucción
<*rm fichero/directorio* >
<*rmdir directorio* >

Cuando se borra un directorio, éste debe estar vacío antes de que se realice el borrado. Una opción es borrar todo el contenido antes o bien podemos dar la opción *-r* al borrar con *rm* de manera que borrará el directorio de manera recursiva.
Salir del directorio carpeta1
cd .. (return) :

Ejecutar la instrucción:
rm -r carpeta1 (return) :

ls -l (return) :

A continuación borraremos el fichero de prueba:
rm prueba (return) :

Con lo que podremos ver que el directorio se queda vacío.
Por último, saldremos del directorio actual mediante
cd .. (return) :
y borraremos también el directorio de la practica0 mediante la misma instrucción *rm* o bien mediante *rmdir*.
rmdir practica0 (return) :
Con ello, podremos ver que hemos eliminado todos los ficheros y directorios que hemos creado y probado durante la práctica, ejecutar (ls o ls -l para comprobarlo):

ls -l (return) :

Estos comandos son los básicos para gestionar ficheros en la *shell* de Linux.

ANEXO I

COMANDOS DE VI

Modo comando

Para terminar la sesión caben varias posibilidades, siempre en modo comando:

:q

Salir cuando no se han hecho modificaciones

:q!

Salir y descartar los cambios

:wq

Salir y guardar los cambios

Modo edición

Cuando se arranca el vi, siempre está en modo comando, por lo que antes de poder escribir texto en el fichero se debe teclear uno de los comandos de entrada del vi , tales como i (insert), para insertar texto en la posición actual del cursor, o a (append) para insertar texto después de la posición actual del cursor.

Para regresar el modo comando, basta con presionar Esc. Si en un momento determinado no se sabe en qué modo se está, simplemente pulsando Esc se asegura uno de que está en modo comando, y se podrá continuar con el trabajo.

Para el resto de comandos, visitar el siguiente tutorial: https://www.dc.fi.udc.es/~afyanez/info-vi/index.html

ANEXO II

VARIABLES DE ENTORNO

Las variables de entorno las necesitamos para guardar algunos valores que pueden ser utilizados cuando trabajamos, modificando valores que por defecto tiene el sistema operativo, es como decorar nuestro lugar de trabajo, además de esto también las usamos en scripts.

El shell tiene sus propias variables, pero nosotros podemos crear todas las que necesitemos.

En el caso de que usemos bash como intérprete de comandos vamos a encontrar que tenemos dos tipos de variables de entorno:

- Variables globales.
- Variables locales.

Una variable global es visible para cualquier proceso en ejecución o que se ejecute desde el shell, en cambio una variable local es aquella que podemos encontrar y ver solo desde shell donde definió.

Variables globales

Siempre que iniciamos el sistema y nos logueamos es el sistema Linux el establece algunas variables de entorno globales, estas variables se escriben LETRAS MAYÚSCULAS para diferenciarlas de los comandos.

Para ver estas variables globales, escribe el comando printenv:

```
$ printenv
SHELL=/bin/bash
LANGUAGE=es_AR:es
PWD=/home/fabian
```

LOGNAME=fabian

XDG_SESSION_TYPE=tty

HOME=/home/fabian

LANG=es_AR.UTF-8

SSH_CONNECTION=192.168.0.222 57076 192.168.0.113 22

XDG_SESSION_CLASS=user

TERM=xterm-256color

USER=fabian

SHLVL=1

XDG_SESSION_ID=1

XDG_RUNTIME_DIR=/run/user/1000

SSH_CLIENT=192.168.0.222 57076 22

PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games

MAIL=/var/mail/fabian

SSH_TTY=/dev/pts/0

_=/usr/bin/printenv

Como vemos en el ejemplo, tenemos muchas variables de entorno globales: Si queremos imprimir solo una de ellas, podemos escribir el comando echo seguido de $NOMBRE_DE_LA_VARIABLE (recordemos que siempre las variables de entorno se escriben con mayúsculas)

Ej: para imprimir la variable HOME escribimos el comando:

echo $HOME

/home/fabian

Variables locales

Nuestro sistema operativo define algunas variables de entorno locales estándar de forma predeterminada estas variables están guardadas en el archivo /etc/profile.

¿Cómo definimos variables locales?

Para definir nuestras propias variables de entorno escribimos en el intérprete de comandos: el nombre de la variable que vamos a crear luego el signo igual y luego el valor que va a tomar esa variable (todo escrito sin espacios), vamos a un ejemplo.

~$ MI_VARIABLE=carreralinux

Si queremos el valor de la variable, use el comando echo, veamos el ejemplo completo.

~$ MI_VARIABLE=carreralinux

~$ echo $MI_VARIABLE

carreralinux

Tenemos que tener en cuenta que esta variable «vive» mientras estemos conectados a este interprete de comandos cuando cerramos la sesión esta variable, es eliminada.

¿Cómo hacemos para que estas variables sean persistentes?

Para declarar una variable de entorno global, debes declarar una variable de entorno local y luego utilizar el comando export de la siguiente manera:

~$ MI_VARIABLE='Soy una variable exportada me transformare en variable global'

~$ echo $MI_VARIABLE

Soy una variable exportada me transformare en variable global

~$ export $MI_VARIABLE

Ahora si salimos del shell y abrimos otra sesión podemos observar que la variable sigue viva esto quiere decir que la variable se transformó en global.