CEU
Fundación San Pablo Andalucía
25º aniversario

SISTEMAS INFORMÁTICOS (DISEÑO APLICACIONES MULTIMEDIA Y DISEÑO DE APLICACIONES WEB)

TEMA 9: INTRODUCCIÓN A LINUX

Profesor: Rafael Madrigal Toscano

TEMA 9: INTRODUCCIÓN A LINUX

1. HISTORIA GENERAL
2. INSTALACIÓN Y GESTIÓN DEL SISTEMA (UBUNTU)
3. ARQUITECTURA DE SISTEMA
4. SHELLS Y EDITORES
5. ESTRUCTURA DE DIRECTORIOS
6. FLUJO DE DATOS
7. PERMISOS
8. ENTORNO DE TRABAJO BASH
9. PROCESOS

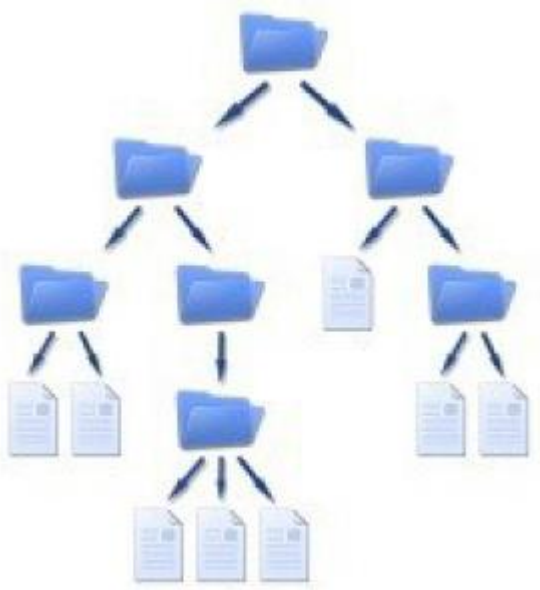
TEMA 9: INTRODUCCIÓN A LINUX

10. DISPOSITIVOS DE ALMACENAMIENTO.MONTAJE Y DESMONTAJE
11. MANEJO DE PAQUETES

TEMA 9: INTRODUCCIÓN A LINUX

5. ESTRUCTURA DE DIRECTORIOS

Es interesante conocer la estructura de directorios en Linux, ya que ello nos permite saber que tipo de información se tiene



Esto nos ayudará para tener una mejor visión de cómo está organizado el sistema operativo GNU/Linux, además con ello sabremos dónde localizar lo que necesitamos.
En la siguiente slide vemos una imagen donde podemos ver una breve descripción de la estructura de directorios de linux:

TEMA 9: INTRODUCCIÓN A LINUX

5. ESTRUCTURA DE DIRECTORIOS

TEMA 9: INTRODUCCIÓN A LINUX

5. ESTRUCTURA DE DIRECTORIOS

/: (Raíz) Es el nivel más alto dentro de la jerarquía de directorios (PRINCIPAL) . De aquí parten los demás directorios y subdirectorios.
/bin: (Binarios) Contiene los binarios básicos, que son los ejecutables del sistema operativo.
/boot: (Arranque) Aquí se encuentran todos aquellos archivos necesarios para que el sistema inicie.
/dev: (Dispositivos) En esta carpeta se encuentran todos los archivos que nos permiten interactuar con los dispositivos hardware de nuestro ordenador. Por ejemplo los usb, sda (o hda) con la información de cada uno de ellos.

/etc: Aquí se guardan los ficheros de configuración de los programas instalados.
/home: (Hogar) Contiene las carpetas por defecto de los usuarios, (sería algo así como el “Documents and Settings” del sistema operativo Windows).
/lib: (Bibliotecas) Contiene las librerías del sistema y los drivers.
/lost+found: (Perdidos y encontrados) información que se guardó de manera incorrecta debido a algún fallo del sistema, se crea en cada partición independientemente.

5. ESTRUCTURA DE DIRECTORIOS

/media: (Media/medios) Directorio donde puede ser utilizado como punto de montaje para las Unidades Extraíbles. Por ejemplo, los dispositivos USB, disqueteras, unidades de CD/DVD.

/mnt: (Montaje) Es un directorio que se suele usar para montajes temporales de unidades. Por ejemplo, Directorios compartidos dentro de una red, alguna partición de Windows, etc.

/opt: (Opcionales) Destinado para guardar/instalar paquetes adicionales de aplicaciones.

/proc: (Procesos) Información de los datos del kernel del sistema de ficheros de Linux, Virtualización y procesos en ejecución.

/root: (Casa del Administrador) Es el directorio /home del administrador. Es el único /home que no está incluido -por defecto- en el directorio HOME.

/sbin (Binarios del Sistema): Son los ejecutables de administración, tales como mount, umount, etc.

5. ESTRUCTURA DE DIRECTORIOS

/srv: (Servicios) En este directorio residen las carpetas accesibles por el programa cliente de un determinado servicio ofrecido por algunos servidores configurados en el sistema. Por ejemplo Apache, ProFtpd, etc.

/sys: (Sistema) Información sobre los dispositivos tal y como los ve el kernel Linux.

/tmp: (Temporales) Es un directorio donde se almacenan ficheros temporales. Cada vez que se inicia el sistema este directorio se borran.

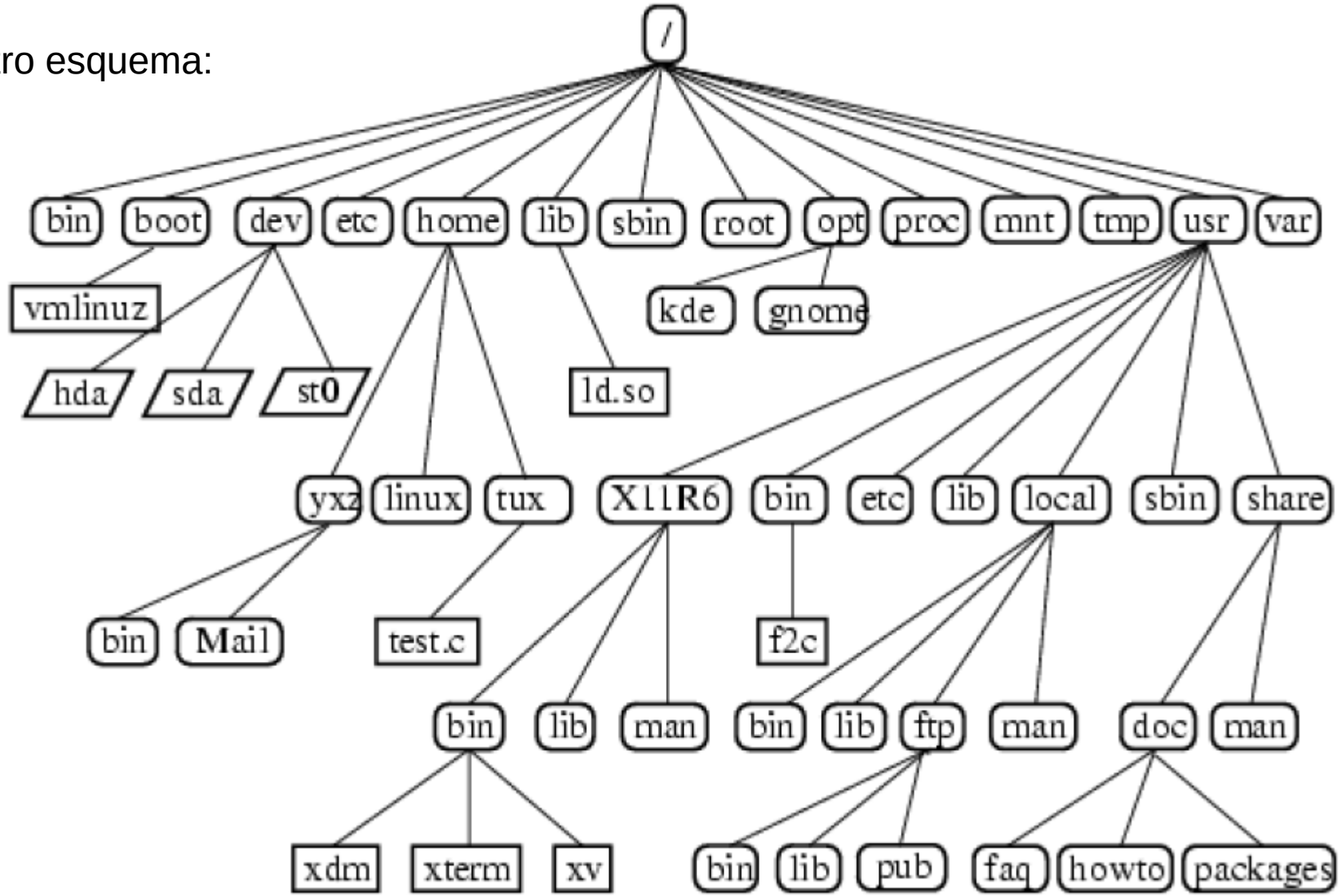
/usr: (Compartido de Usuarios) Es el directorio padre de otros subdirectorios de importancia. Se encuentran la mayoría de los archivos del sistema, aplicaciones, librerías, manuales, juegos… Es un espacio compartido por todos los usuarios del sistema.

/var: (Variables) Ficheros y datos que cambian frecuentemente (logs, bases de datos, colas de impresión…)

TEMA 9: INTRODUCCIÓN A LINUX

5. ESTRUCTURA DE DIRECTORIOS

Otro esquema:



TEMA 9: INTRODUCCIÓN A LINUX

5. ESTRUCTURA DE DIRECTORIOS

Dentro de la categorización de los directorios encontramos:

- **Estáticos:** Contiene archivos binarios, bibliotecas, y otros archivos están en Read Only (Solo lectura) que no cambian sin la intervención del administrador.
  /bin, /sbin, /boot, /opt.

- **Dinámicos:** Son aquellos que los archivos dentro de estos van cambiando. Generalmente se encuentra como Read-Write (Lectura-Escritura).
  /var/spool, /var/lock, /var/mail, /home.

- **Compatibles:** Se pueden encontrar archivos comunes que van a estar en cualquier distribución.
  /usr/bin, /opt

- **No Compatibles:**
  Contiene archivos que no son compatibles con otras distribuciones.
  /etc, /boot, /var/run, /var/lock.

TEMA 9: INTRODUCCIÓN A LINUX

6. FLUJO DE DATOS

En Linux al igual que en Unix todos los procesos (programas en ejecución) tienen asociados tres flujos (streams) de datos principales. Estos son:

1. La entrada estándar. (stdin) Es donde un proceso puede tomar los datos que maneja y que no se indican mediante argumentos u opciones. Por defecto se toma a partir del teclado.
2. La salida estándar. (stdout) Es donde un proceso escribe los resultados de su ejecución. Por defecto es la terminal (pantalla) donde se invocó el programa correspondiente.
3. La salida de errores. (stderr) Es donde un proceso escribe los posibles errores durante su ejecución. Por defecto es la terminal (pantalla) donde se invocó el programa correspondiente

Los flujos de datos se almacenan en descriptores de ficheros que se identifican por un número en la forma siguiente:

0: representa la entrada estándar.
1: representa la salida estándar.
2: representa la salida de errores.

TEMA 9: INTRODUCCIÓN A LINUX

6. FLUJO DE DATOS

¿Qué es bash?

GNU Bash o simplemente Bash es una popular interfaz de usuario de línea de comandos, específicamente un shell de Unix; así como un lenguaje de scripting. Bash fue originalmente escrito por Brian Fox para el sistema operativo GNU, y pretendía ser el reemplazo de software libre del shell Bourne.

En bash, al igual que en otros shells, los flujos de datos se pueden redireccionar libremente hacia distintos ficheros. En esencia este mecanismo consiste en que la salida de un proceso (estándar o de errores) puede escribirse en un fichero en lugar de la terminal asociada, así como la entrada puede tomarse también a partir de un fichero en lugar de utilizar lo escrito mediante el teclado.

Para indicar un redireccionamiento se utilizan los signos de comparación < y >. De esta forma se generan dos tipos de construcción:

- Instrucción > salida: indica el redireccionamiento del flujo de datos de la instrucción al fichero nombrado salida.
- Instrucción < entrada: indica el redireccionamiento del contenido del fichero nombrado entrada como flujo de datos de entrada para la instrucción.
- Instrucción 2> salida: indica el redireccionamiento del flujo de datos DE LOS ERRORES de la instrucción al fichero nombrado salida.

TEMA 9: INTRODUCCIÓN A LINUX

6. FLUJO DE DATOS (REDIRECCIONES)

Ejemplos: # cat /etc/shadow > cont	Se redirecciona la salida estándar hacia un fichero con nombre cont
$ write pepe < msj.txt	Toma la entrada en lugar de desde el teclado, desde el fichero msj.txt
$ cat > teléfonos	Toma de la entrada estándar (teclado) y lo envía a la salida que será el fichero teléfonos. (Terminar la entrada con Ctrl D).
$ ls > /dev/null	Se redirecciona la salida estándar al dispositivo especial de caracteres /dev/null provocando su desaparición.
$ ls /tmp 2> fich-errores	Se redirecciona la salida de errores hacia un fichero fich-errores.
$ find / -iname “carta” > sa 2> err	Ejecuta el comando find, mandando la salida normal de la orden al fichero sa y mandando la salida de errores al fichero err.

TEMA 9: INTRODUCCIÓN A LINUX

6. FLUJO DE DATOS

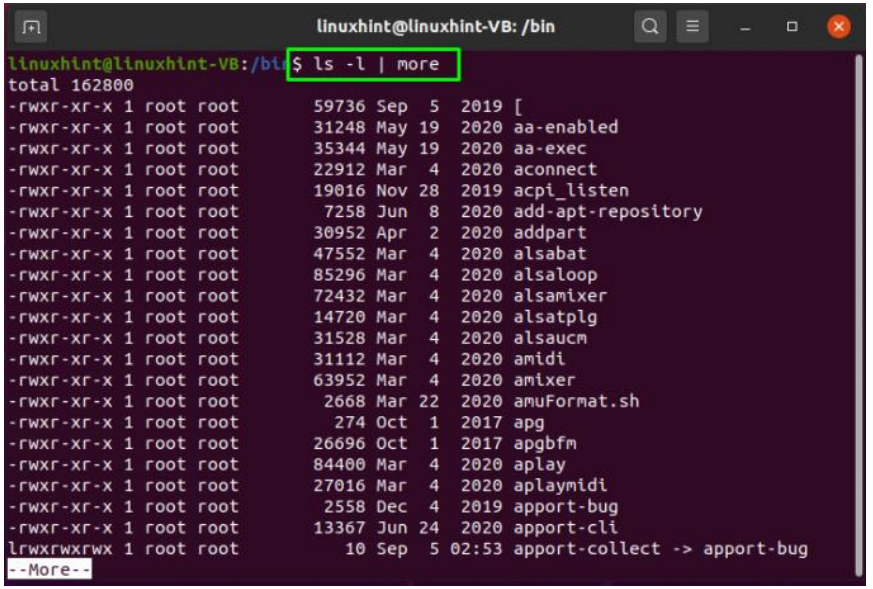
Siempre que se emplee la forma [x]>salida, si el fichero salida existe se sobrescribirá y si no, se creará. Para añadirle algo más a su contenido anterior (append) en lugar de sobrescribirlo, se puede emplear >>.

$echo “Hola” >> fichero

Se redirecciona el texto Hola hacia fichero. Si fichero ya existe se añadirá el texto al final, si no existe, se creará fichero y luego se le añadirá el texto.

¿Qué son la **tuberías** o pipes?

El concepto de tubería o pipe en linux consiste en unir dos comandos en la terminal, con lo cual podemos tomar la salida del primer comando y utilizarlo como entrada en el segundo, dando lugar a que la línea de comandos sea todavía más poderosa, gracias a las innumerables combinaciones que se podrían hacer.



TEMA 9: INTRODUCCIÓN A LINUX

6. FLUJO DE DATOS

Muchos de los comandos mencionados anteriormente, que reciben como argumento un fichero, en caso de omitirse este, utilizan su entrada estándar. Esta entrada puede provenir a su vez de la salida de otros comandos. Gracias a esto las tuberías permiten realizar filtrados de los más diversos tipos.

Las tuberías pueden estar formadas por un número ``ilimitado'' de comandos. Estos no se ejecutan secuencialmente, o sea no se espera a que termine uno para ejecutar el siguiente, sino que se va haciendo de forma concurrente. El carácter que se emplea para separar un comando de otro mediante una tubería es |. (Alt Gr + 1 ó Alt + 124)

Ejemplos:

$ ls -l /dev | less	Toma la salida de ls y la envía como entrada a la orden less, con lo que se consigue un listado paginado.
$ ls -R /etc | sort | more | uniq	Lista recursivamente el contenido del directorio /etc y ordena la salida paginándola y mostrando únicamente las líneas únicas. (Sin líneas duplicadas).

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

GNU/Linux, al ser un sistema diseñado fundamentalmente para trabajo en red, la seguridad de la información que almacenemos en nuestros equipos (y no se diga en los servidores) es fundamental un sistema con permisos sobre directorios y ficheros, ya que muchos usuarios tendrán o podrán tener acceso a parte de los recursos de software (tanto aplicaciones como información) y hardware que están gestionados en estos ordenadores.
(Nótese que en GNU/Linux hablamos de **directorios** y no de carpetas.)

Lo siento mucho por mis amigos “Guindoleros” pero ese sistema “malévolo” no fue pensado originalmente como multiusuario y de ahí que Linux lo tenga de forma nativa muy bien implementado.

En GNU/Linux, los permisos o derechos que los usuarios pueden tener sobre determinados archivos contenidos en él se establecen en tres niveles claramente diferenciados.
Estos tres niveles son los siguientes:

1. Permisos del propietario. (U- User)
2. Permisos del grupo. (G-Group)
3. Permisos del resto de usuarios (o también llamados “los otros”). (O-Other)

Nota: Permisos para todos. (A-All)

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

Para tener claros estos conceptos, en los sistemas en red (como lo es el pingüino) siempre existe la figura del administrador, superusuario o root. Este administrador es el encargado de crear y dar de baja a usuarios, así como también, de establecer los privilegios que cada uno de ellos tendrá en el sistema. Estos privilegios se establecen tanto para el directorio HOME de cada usuario como para los directorios y archivos a los que el administrador decida que el usuario pueda acceder.
Para cada uno de los tres grupos mencionado anteriormente de usuarios (U,G,O) existen tres tipos de permisos fundamentales:

- ✓ r: read (lectura). El usuario que tenga este permiso podrá si es un directorio, listar los recursos almacenados en él, y si es cualquier otro tipo de fichero podrá leer su contenido.
- ✓ w: write (escritura). Todo usuario que posea este permiso para un fichero podrá modificarlo. Si se posee para un directorio se podrán crear y borrar ficheros en su interior.
- ✓ x: execute (ejecución). Este permiso para el caso de los ficheros permitirá ejecutarlos desde la línea de comandos y para los directorios, el usuario que lo posea tendrá acceso para realizar el resto de las funciones permitidas mediante los otros permisos (lectura y/o escritura).

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

Evidentemente, para realizar operaciones sobre cualquier archivo será necesario tener permiso de ejecución.

Para acceder a un recurso de cualquier forma (ejecución, lectura o escritura) se deben tener permisos de ejecución para todos los directorios que contienen al recurso directa e indirectamente.
Los tres tipos de permisos mencionados poseen una representación numérica basada en el sistema octal que parte de representar como ``1'' los bits de los permisos otorgados y``0'' para los negados. Luego se transforma la representación binaria así obtenida en octal.
Los permisos siempre van formando tríos, de la forma rwx. De esta forma se obtienen para cada tipo de permiso los siguientes valores:

Permiso	r	w	x
Valor	4	2	1

r = 100 (4 en octal) (r--)

w = 010 (2 en octal) (-w-)

x = 001 (1 en octal) (--x)

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

La combinación de los tres tipos de permisos para un tipo de usuario oscila desde cero (ningún permiso) hasta siete (todos los permisos).

r	w	x	Binario	Decimal	Permisos
r	w	-	110	6	Lectura y escritura, no ejecución.
r	-	x	101	5	Lectura y ejecución.
r	-	-	100	4	Solo lectura.
-	-	-	000	0	Ningún permiso.

Los permisos ``totales'' de un recurso constan de nueve indicadores, donde los tres primeros indican los permisos asociados al **usuario** dueño, los otros tres al **grupo** y los últimos 3 a los **otros**, al resto de los usuarios.

Permisos (ugo)			Valor octal	Permisos al usuario	Permisos al grupo	Permisos a otros
rw-	rw-	rw-	6 6 6	Lectura y escritura	Lectura y escritura	Lectura y escritura
rwx	rwx	---	7 7 0	Todos	Todos	Ninguno
rw-	r--	r--	6 4 4	Lectura y escritura	Lectura	Lectura
rwx	r-x	---	7 5 0	Todos	Lectura y ejecución	Ninguno
r--	---	---	4 0 0	Lectura	Ninguno	Ninguno

Sólo el dueño de un recurso tendrá derecho a cambiar sus permisos, además del root por supuesto.

7. PERMISOS

Además existen los permisos s y S y t y T que son mas raramente usados, de hecho algunas distros no los tienes porque son inseguros de por si, por lo que es muy recomendable no usarlos a menos que sea estrictamente necesario.

* Utilice el comando ls para listar un archivo y observe sus permisos.

El primer carácter indica el tipo de recurso, y puede ser:

- d : directorio
- l : enlace
- b : dispositivo de bloque
- c : dispositivo de caracteres
- s : socket
- p: tubería (pipe)
- - : fichero regular

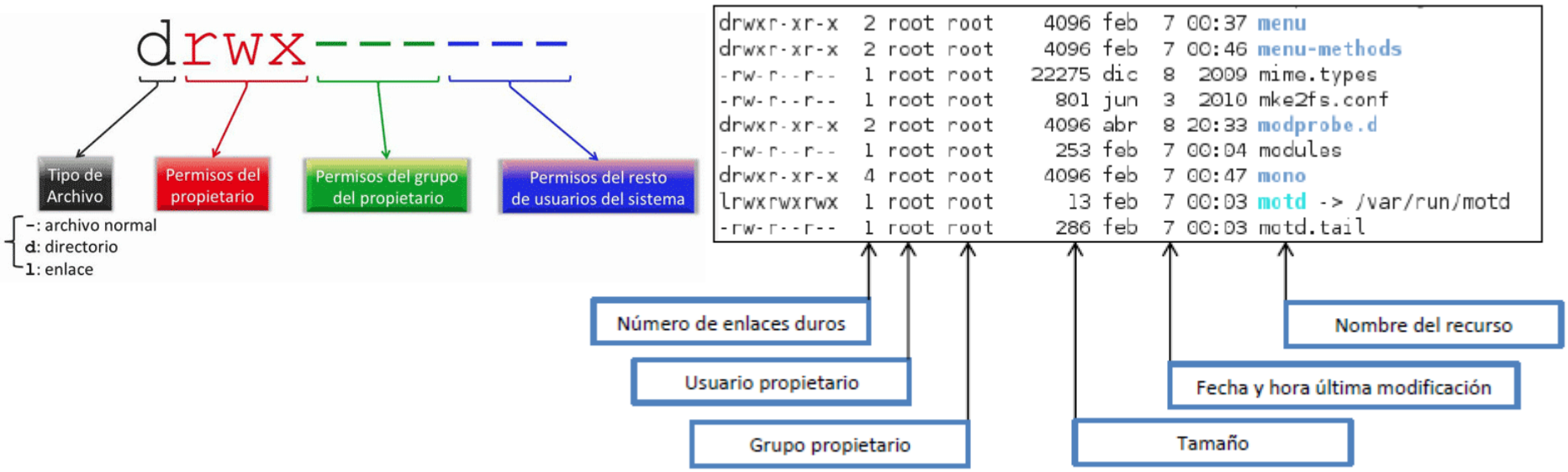
TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

Los caracteres 2,3 y 4 indican los permisos para el usuario dueño del recurso.

Los caracteres 5,6 y 7 indican los permisos para el grupo dueño del recurso.

Los caracteres 8,9 y 10 indican los permisos para el resto de usuarios, es decir, los usuarios que no son



TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

El resto de las columnas de la salida representan para cada elemento:

- El número de enlaces duros que posee. En el caso de un directorio, indica la cantidad de subdirectorios que contiene contando a los directorios especiales ``.'' y ``..''
- El identificador del dueño.
- El identificador del grupo.
- El tamaño en bytes si es un fichero y si es un directorio el tamaño en bloques que ocupan los ficheros contenidos en él.
- La fecha y hora de la última modificación. Si la fecha es seis meses antes o una hora después de la fecha del sistema se coloca el año en lugar de la hora.
- El nombre del recurso.

Para cambiar los permisos de un recurso se utiliza el comando chmod.

Sintaxis: chmod [opciones] <permisos> <ficheros>
Las formas de expresar los nuevos permisos son diversas, se pueden usar números o caracteres para indicar los permisos. Podremos comprender mejor cómo funciona la orden mirando directamente algunos ejemplos:

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

$ chmod u+x clase.txt	Añade el permiso de ejecución (+x) al usuario dueño (u) del fichero clase.txt.
$ chmod g=rx program.sh	Asigna exactamente los permisos de lectura y ejecución (rx) al grupo (g) sobre el fichero program.sh.
$ chmod go-w profile	Elimina el permiso de escritura (-w) en el grupo y en otros (go) del fichero o directorio profile.
$ chmod a+r,o-x *.ts	Añade el permiso de lectura (+r) para todos los usuarios (a) y elimina el de ejecución (-x) para otros (o) en todos los ficheros terminados en .ts.
$ chmod +t tmp/	Añade el permiso especial t al directorio tmp.
$ chmod 755 /home/pepe/doc/	Asigna los permisos con representación octal 755 (rwx r-x r-x) al fichero /home/pepe/doc.
$ chmod -R o+r apps/	Añade el permiso de lectura a otros en el directorio apps y además lo hace de forma recursiva, añadiendo dicho permiso también en todos los ficheros y directorios contenidos en apps.
# chmod 4511 /usr/bin/passwd	Asigna los permisos con representación octal 4511 (r-s--x--x)
$ chmod 644 *	r w – r - - r - - Lectura y escritura para el usuario, lectura para el grupo y lectura para otros.

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

Ejercicio

Como ejercicio, cread un directorio copia en vuestro home de usuario, y dentro copiad todos los ficheros que existen en el directorio /etc. Modificad los permisos de los ficheros copiados, probando tanto la orden chmod de forma numérica como mediante caracteres.

Si creamos un nuevo fichero, veremos como es creado con unos permisos por defecto, normalmente 644.

Para determinar estos permisos que se asocian por defecto a los ficheros o directorios creados, cada usuario posee una máscara de permisos. Esta mascara por defecto suele ser 022 para los usuarios comunes.

Para calcular los permisos finales conociendo la máscara, se hace la siguiente operación por parte del sistema:

TEMA 9: INTRODUCCIÓN A LINUX

7. PERMISOS

Tipo de ficheros	Operación	Desarrollo	Resultado
Ficheros normales	666 - máscara	(666 – 022 = 644)	644
Directorios y Ficheros ejecutables	777 - máscara	(777 – 022 = 755)	755

Para ajustar la máscara se puede emplear el comando umask.
Sintaxis: umask [-S] [máscara]

Ejemplos:

Como vemos, si usamos el formato numérico en umask, tendremos que restar la máscara al total de permisos para saber que permisos se asignarán por parte del sistema. Sin embargo, si usamos el formato simbólico (-S) de umask, asignaremos directamente los permisos que el sistema asignará. Para dejar este umask fijo, conviene agregarlo al fichero .bash_profile o al fichero .bash_rc de nuestro directorio de inicio.

$ umask	Sin argumentos muestra la máscara actual en formato numérico.
$ umask -S	muestra el complemento de la máscara en formato de caracteres
$ umask 037	asigna la máscara 037
$ umask –S u=rwx,g=rwx,o=rx	Directorios y archivos ejecutables se crearán con los permisos 775 y los archivos comunes con los permisos 664.
$ umask 077	Los nuevos directorios tendrán el permiso 700 y los nuevos ficheros tendrán el permiso 600.

MUCHAS GRACIAS