Repaso principales comandos UNIX

Ángel del Río Álvarez

Sistemas Operativos 2019/2020

El intérprete de comandos

- Prompt
  - Cadena que se muestra indicando que el intérprete está listo para recibir comandos.
- Shell
  - Tipo de intérprete de comandos
  - Por defecto: `/bin/bash`
    - Historial de órdenes, autocomplección, colores, etc..
    - Prompt:
      - **`nombre_usuario@nombre_maquina:directorio_actual$`**
      - El directorio personal del usuario se representa con `~`
  - Otros Shell (características parecidas):
    - /bin/sh
    - /bin/ksh

Sintaxis de los comandos

- `comando [-opciones] [argumentos]`
  - [ ] → opcionalidad
  - Las opciones modifican el modo en que se comporta el comando.
  - Case sensitive (distingo mayúsculas y minúsculas)
  - Puedo introducir más de una orden en la misma línea usando “;”

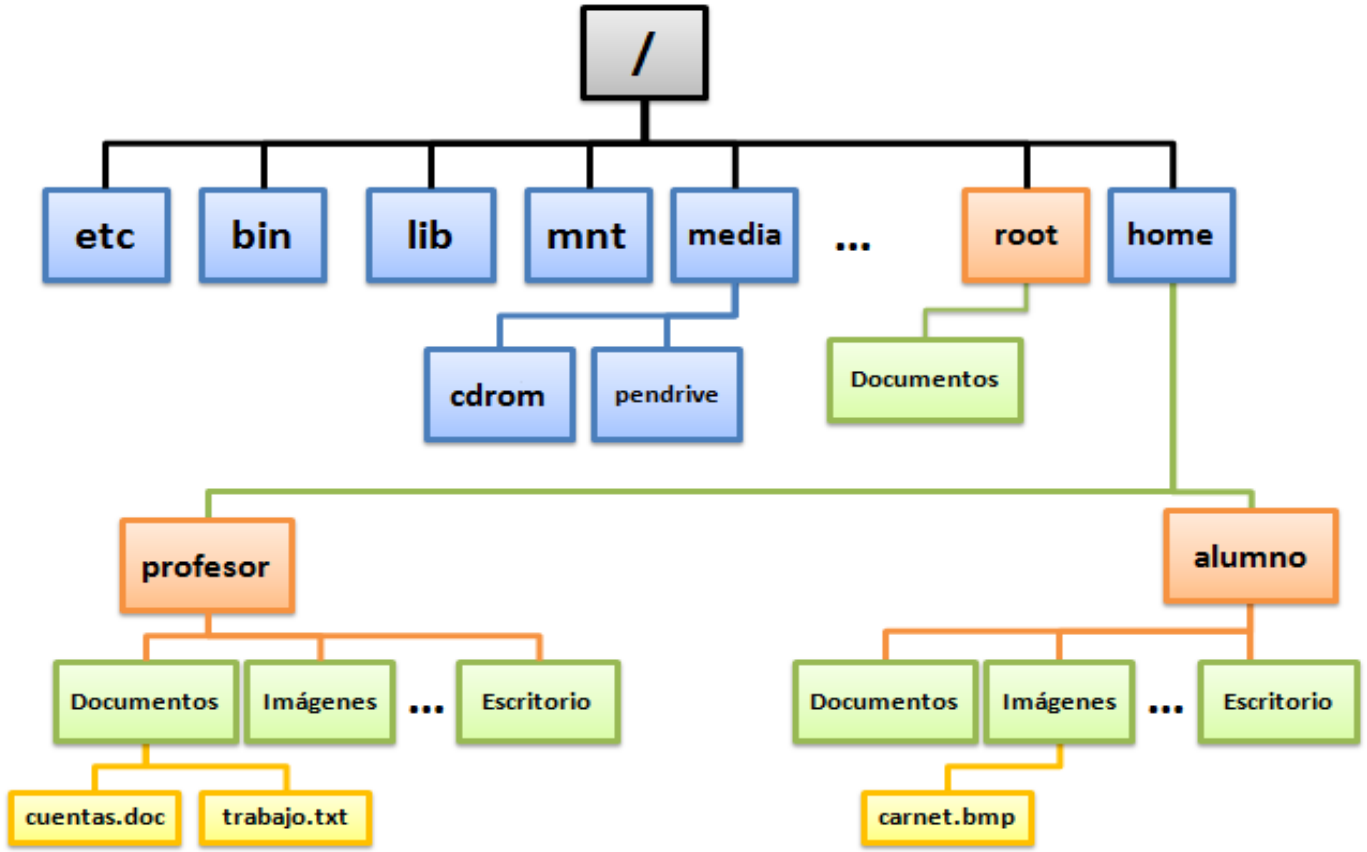
Ayuda

- Ayuda
  - ***$ man comando → IMPRESCINDIBLE!!***
    - q para salir
  - `$ comando --help`
  - `$ whatis`
- Borrar la pantalla
  - `$ clear`
- **Tabulador**: Autocomplección!

Comandos para el manejo de directorios

Principales comandos Linux

FHS (File Hierarchy System)



FHS (File Hierarchy System)

/home

Directorios personales de los usuarios locales. Para cada usuario, se creará un subdirectorio con el nombre del usuario donde colgarán sus documentos personales (música, imágenes, escritorio, etc…)

/root

Directorios personales del **administrador**

/dev

Archivos de dispositivo (guardan info sobre el hardware)

/media

Es el directorio donde se montan los **soportes extraíbles** como CDs, cámaras digitales, etc.

/mnt

Sistemas de ficheros montados.

/tmp

Archivos temporales.

/etc

Ficheros de **configuración del sistema,** scripts (secuencias de comandos) de inicio, etc.

/lib

Bibliotecas del sistema

/sbin

Archivos binarios importantes para el **sistema.**

/bin

Aplicaciones binarias importantes.

/usr

Aplicaciones y archivos disponibles para todos los usuarios.

Rutas absolutas y relativas

- Rutas absolutas
  - Camino completo desde /
- Rutas relativas
  - Camino desde el directorio actual
  - Directorios especiales:
    - . (punto): El propio directorio
    - .. (punto punto). El padre del directorio

Desplazamiento por el árbol de directorios: Comandos I

- **`$ pwd`**
  - Muestra el directorio actual
- **`$ ls [opciones][ruta]`**
  - Muestra el contenido del directorio
  - Si no especifico nada muestra el contenido del directorio actual
  - Opciones (ver ayuda)
    - `-l` Formato largo
    - `-a` Todos los archivos
      - Incluye archivos ocultos (comienzan por `.`)

Ejemplos

- **`$ ls`**
  - Muestra el contenido del directorio actual
- **`$ ls -a`**
  - Muestra todos los archivos del directorio actual
- **`$ ls -la`**
  - Muestra todos los archivos del directorio actual en formato largo
- **`$ ls -la /home/Justino`**
  - Muestra todos los archivos del directorio `/home/Justino` en formato largo

Desplazamiento por el árbol de directorios: Comandos II

- **`$ cd path`**
  - Cambia de directorio
    - Si no pongo nada cambia a la carpeta personal del usuario (HOME)
  - **`path`** puede ser una ruta absoluta o relativa
  - Ejemplos
    - **`cd`**
      - Cambia al directorio HOME
    - **`cd ..`**
      - Cambia al padre
    - **`cd /etc`**
      - Cambia al directorio /etc

Crear un directorio

- **$ mkdir [-p] [path]/nombre_dir**
  - -p: Crea los directorios intermedios
  - Se puede crear más de un directorio a la vez
  - Ejemplos:
    - `$ mkdir –p /home/naranco/som /home/naranco/mme`
      - Crea los directorios `/home/naranco/som` y `/home/naranco/mme`
    - `$ mkdir –p /home/naranco/som mme ofimatica`
      - Crea el directorio `/home/naranco/som` y los directorios mme y ofimatica dentro del directorio donde estamos
    - `$ mkdir /home/naranco/”Sistemas Operativos Monos”`
      - Crea el directorio `Sistemas Operativos Monos` dentro de `/home/naranco`. Si no existía `/home/naranco` no hace nada.

Borrar un directorio

- **`$ rmdir [opciones][path]/nombre_dir`**
  - Borra directorios **vacíos**.
  - Opciones:
    - -p: Si especifico un directorio anidado, se borran también los padres.
  - Ejemplo:
    - `$ rmdir tema1 tema2 tema3`
      - Borra los directorios tema1, tema2 y tema3 del directorio actual.
    - `$ rmdir –p APUNTES/TEMA1/Transparencias`
      - Borra los tres directorios anidados (si están vacíos),

Comandos para visualizar contenido de los ficheros

Principales comandos Linux

Ficheros: Mostrar contenido

- **$ cat fichero**
  - Muestra el contenido del fichero
- **$ more fichero**
  - Muestra el contenido del fichero pantalla a pantalla.
- **$ less fichero**
  - Muestra el contenido del fichero permitiendo retroceso y avance.
- **$ sort fichero**
  - Ordena las líneas de los ficheros de entrada. Por defecto toma el espacio en blanco como separador de campo.

Comandos para el manejo de ficheros

Principales comandos Linux

Editar un fichero de texto

- Algunos editores
  - vi
    - Uso complejo pero muy potente
    - Usado en la gran mayoría de las distribuciones
  - nano
    - Uso algo más sencillo
    - Distribuciones basadas en Debian
  - gedit
    - Distribuciones basadas en GNOME
    - Modo gráfico
  - Ejemplos de uso:
    - `$ vi ladrillo.txt`
    - `$ nano ladrillo.txt`
    - `$ gedit ladrillo.txt`

SI LOS ARCHIVOS NO EXISTEN SE CREAN

Creación y borrado de ficheros

- **$ touch [opciones] nombre_archivo**
  - Sin opciones crea un fichero vacío
  - Ejemplo:
    - `$ touch ladrillo.txt`
- **$ rm [opciones] fichero/s**
  - -r: Borra recursivamente. Se usa para directorios.
  - -i: Pide confirmación antes de ir borrando
  - Ejemplo:
    - `$ rm –r temp`
      - Borra temp y sus subdirectorios

Copiar ficheros

- **`$ cp [opciones] fichero1 fichero2`**
  - Copia fichero1 en otro llamado fichero2
  - Opciones:
    - -r: Copia recursivamente (subdirectorios)
  - Ejemplo:
    - `$ cp –r cosas  /backups`
      - Copia dir (y todos sus subdirectorios) en el directorio /backups
      - Si no existe lo crea

Mover/renombrar ficheros

- mv se utiliza con 2 funciones distintas:
  - **`$ mv [opciones] fichero1 directorio_destino`**
    - Mueve el/los archivos indicados al directorio destino
    - Ejemplo:
      - `$mv config.txt /etc/app/`
  - **`$ mv [opciones] fichero1 fichero2`**
    - Renombra fichero1 como fichero2
    - Ejemplo:
      - `$mv config.txt config.txt.bak`

Ejercicio

- En un directorio “Ejemplos” dentro de nuestro espacio de usuario:
  1. Crear dos subdirectorios “ejemplos_a” y “ejemplos_b”
  2. Dentro de ejemplos_a crear un documento de texto llamado “texto_a.txt” con el siguiente texto: “Este es un documento de prueba que está dentro del directorio A”
  3. Copiar texto_a.txt a ejemplos_b y modificar su texto indicando que se encuentra en el directorio B.

Comandos para la búsqueda y recorte de texto en ficheros

Principales comandos Linux

Búsqueda de texto en un archivo

- `$ grep [opciones] cadena_texto ficheros`
  - Busca una cadena de texto en un archivo.
  - Opciones:
    - -c → Muestra el número de línea
    - -i → Ignora mayúsculas/minúsculas
  - Ejemplo:
    - `$ grep  bash  /etc/shells`
      - Busca la palabra bash en el archivo /etc/shells (donde se guardan todos los shells)

Cortar texto en un archivo

- **`$ cut opciones [fichero]`**
  - Obtiene información de un fichero.
  - Muy relacionado con ficheros de datos, en los que hay un delimitador de campo y el salto de línea separa los registros
  - Opciones:
    - -d → Especifica que carácter hace de delimitador de campos
    - -f → Campos que se quieren seleccionar
  - Ejemplo:
    - `$ cut –f 1,5 –d ‘:’ /etc/passwd`
      - Campos 1 y 5 del /etc/passwd utilizando el ‘:’ como separador
    - `$ cut –f 1-3,5 –d ‘:’ /etc/passwd`
      - Campos del 1 al 3, y 5

Cortar texto en un archivo

- **`$ head/tail [opciones] [fichero]`**
  - Obtiene las n primeras (head) o últimas (tail) líneas de un fichero
  - Por defecto n=10
  - Opciones:
    - -n Número de líneas
  - Ejemplo:
    - `$ head /etc/passwd`
      - Muestra las 10 primeras líneas de /etc/passwd
    - `$ tail –n 5 /etc/passwd`
      - Muestra las 10 últimas líneas de /etc/passwd

Otros comandos importantes

Principales comandos Linux

Imprimir texto en pantalla: echo

- **$ echo [opciones] cadena**
  - Muestra la cadena
  - Opciones
    - n No introduce un salto de línea al final
    - E Interpreta caracteres especiales (\a, \b, \n, …)
  - Ejemplos
    - Echo Hola Mundo
      - Muestra “Hola Mundo”

Mostrar la fecha: date

- **$ date [opciones] [+formato]**
  - Muestra la fecha actual en el formato deseado
  - Formato
    - %d, %m, %y → Día, mes y año (numéricos)
    - %A → Día de la semana
    - %B → Mes (texto)
  - Ejemplos
    - date + '%d/%m/%y'
      - Muestra "12/01/2020"
    - date + Hoy es %A, %d de %B, del año %y'
      - Muestra "Hoy es 12 de Enero del año 2020"

Mostrar información sobre procesos: ps

- **`$ ps [opciones]`**
  - Muestra los procesos en ejecución
  - Opción –A: Muestra los procesos de todos los usuarios
  - Ejemplo:
    - `$ ps`
      - Muestra los procesos que se ejecutan
    - `$ ps -A | grep firefox`
      - Muestra el proceso firefox

Finalizar un proceso: Kill

- **`$ kill [opciones] pid`**
  - Finaliza (mata) un proceso
  - Opción -9: Fuerza la finalización
  - Debemos averiguar el pid previamente con ps

Conteos sobre un fichero: wc

- **`$ wc [opciones] fichero`**
  - Opciones
    - -l número de líneas
    - -c número de bytes
    - -m número de caracteres
    - -L longitud de la línea más larga
    - -w número de palabras
  - Ejemplos
    - wc –w texto.txt
      - Devuelve el número de palabras del fichero texto.txt

Búsqueda de un fichero: find

- **`$ find directorio [opciones] [acción]`**
  - La búsqueda se realiza en directorio
  - La búsqueda se realiza por el criterio indicado en opciones:
    - name nombre: Por nombre
    - user usuario: Por usuario
  - Ejemplos:
    - `$ find ./ -name test`
      - Busca en el directorio donde me encuentro el archivo llamado test.
    - `$ find / -name pajaro`
      - Busca en todo el sistema todos los archivos llamados pajaro.
    - `$ find /home –user pepito`
      - Busca en `/home` todos los archivos del usuario pepito

Metacaracteres

- Nos permiten referirnos a un conjunto de ficheros o directorios fácilmente:
  - ? → Representa un único carácter
  - * → Representa cualquier carácter
  - [ ] → Representa únicamente los caracteres ubicados entre los corchetes
  - [!] → Representa cualquier carácter que no esté entre los corchetes

Ejercicio

- Busca en /home todos los archivos que comienzan por p
- Muestra todos los archivos de 3 letras con extensión txt
- Borra los archivos cuya primera letra esté entre a y d
- Mueve todos los archivos de texto que no comienzan por a la carpeta indicada
- Borra todo lo que hay dentro de Ejemplo1 (sin incluirlo)

Solución

- `$ find /home –name p*`
  - Busca en /home todos los archivos que comienzan por p
- `$ ls ???.txt`
  - Muestra todos los archivos de 3 letras con extensión txt
- `$ rm [a-d]*`
  - Borra los archivos cuya primera letra esté entre a y d

Soución (2)

- `$ mv [!a]*.txt /home/pepe/Documentos`
  - Mueve todos los archivos de texto que no comienzan por a la carpeta indicada
- `$ rm -r Ejemplo1/*`
  - Borra todo lo que hay dentro de Ejemplo1 (sin incluirlo)

Modelo de permisos

- Los permisos de acceso de cada fichero o directorio se establecen en tres niveles:
  - El de usuario que ha creado un fichero (*propietario*) (**u**)
  - El del grupo al que pertenece el usuario (*grupo)* (**g**)
  - El de todos los demás usuarios del sistema (*otros*) (**o**)

Acceso

- **Para cada nivel** se establecen 3 tipos de acceso:
  - Lectura (r)
    - Fichero: Permite ver su contenido
    - Directorio: Permite listar su contenido (ls)
  - Escritura (w)
    - Ficheros: Permite modificar su contenido
    - Directorios: Permite crear/borrar ficheros y subdirectorios
  - Ejecución (x)
    - Ficheros: Permite ejecutarlo
    - Directorios: Permite acceder a él (cd)

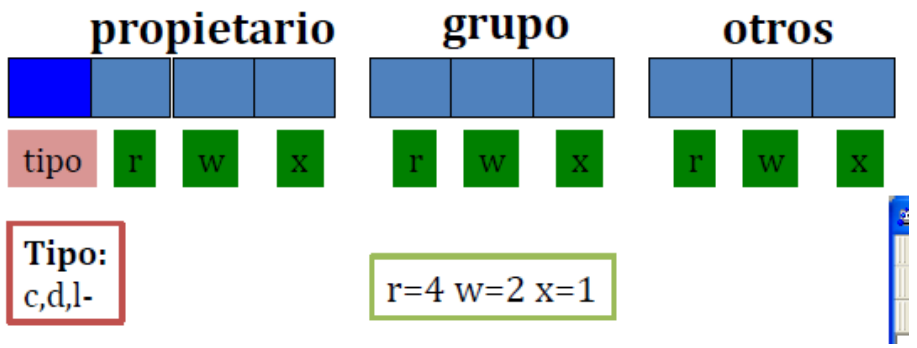
Establecer permisos-I

- Los permisos se representan mediante 3 números en octal (0-7)
  - Ej: 755
    - El número más a la izquierda representa los permisos asignados al usuario
    - El número más central representa los permisos asignados al grupo
    - El número más a la derecha representa los permisos asignados al resto

Establecer permisos-II

- Significado del número octal
  - Si lo pasamos a binario, cada uno de los bits se corresponden a los permisos de lectura, escritura y ejecución (r,w,x) respectivamente:

Decimal	Octal	Permisos
0	000	---
1	001	--x
2	010	-w-
3	011	-wx
4	100	r--
5	101	r-x
6	110	rw-
7	111	rwx



Establecer permisos-III

- **`$ chmod permisos fichero/s`**
  - Permisos. Pueden asignarse en 2 formatos
    - Octal: 3 dígitos en octal
    - Simbólicos: Especifico si añado (+) o quito (-) el permiso (r,w,x) a usuario, grupo u otros (u,g,o)
  - Ejemplos:
    - `$ chmod 755 números.txt`
      - Establece todos los permisos para propietario y lectura y ejecución para grupo y resto
    - `$ chmod go+rw`
      - Añado permisos de lectura y ejecución al grupo y a los otros

Cambiar permisos

- **`$ chown nuevo_propietario archivo`**
  - Establece el propietario del archivo
  - Ejemplo:
    - `$ chown pepito archivo`
- **`$ chgrp nuevo_grupo archivo`**
  - Cambia el grupo del archivo

Redirección de E/S

Principales comandos Linux

Redirección de E/S

- Redirección de salida
  - comando > fichero
    - La salida del comando es la entrada del fichero. Si el fichero existe se sobrescribe
  - comando >> fichero
    - La salida del comando es la entrada del fichero. Si el fichero existe se añade al final
- Redirección de entrada
  - comando < fichero
    - Fichero es la entrada del comando
- Tuberías o pipes
  - comando1 | comando2
    - La salida de comando1 es la entrada de comando2
  - Comando 1 | fichero

Redirección de E/S: Ejemplos

- `$ ls –la > listado.txt`
  - Escribe en el fichero `listado.txt` el listado del directorio actual
- `$ grep Pepe < nombres.txt`
  - Busca la cadena `Pepe` en el fichero `nombres.txt`
- `$ ls /home | grep juan`
  - Enlaza la salida del comando `ls` con la entrada de `grep`
  - Sólo muestra las líneas del listado que contienen la palabra `juan`

Ventajas *pipes* sobre redirecciones a fichero

- Permiten comunicar N programas de forma concurrente (se ejecutan todos a la vez). Mediante redirecciones, se ejecutan en serie
  - Programa1 [< fichero_entrada] | Programa2 | Programa3 | … | ProgramaN [> fichero_salida]
- No utilizan memoria secundaria, solo RAM que es mucho más rápida. Redireccionando a un fichero, este se crea y guarda en memoria secundaria
- No es necesario inventarse nombres de ficheros auxiliares para realizar la comunicación ni eliminarlos luego