CURSO LINUX ADMINISTRATOR

Comandos de Administración

- INSTRUCTOR: RUDY SALVATIERRA RODRIGUEZ

Búsqueda de archivos

Tipos de Rutas

- **Rutas absolutas:** Son aquellas que se contruyen poniendo toda la ruta hasta el directorio que nos queremos mover.

  Como ejemplo si tenemos la siguiente estructura de directorios:

  /directorio1/directorio2/directorio3

  Si estamos en el directorio2 y queremos ir al directorio3 se pondría:
  cd /directorio1/directorio2/directorio3 ruta completa

- **Relativas: Rutas relativas:** Son aquellas que se contruyen a partir del directorio en el que nos encontramos.

  Como ejemplo si tenemos la siguiente estructura de directorios:

  /directorio1/directorio2/directorio3

  Si estamos en el directorio2 y queremos ir al directorio3 se pondría:
  cd directorio3

Búsqueda de archivos

- **Caracteres especiales**
  - *****: sustituye una cantidad cualquiera de caracteres
  - **?**: sustituye exactamente un carácter
  - **[]**: sustituye un carácter de los indicados
- **Ejemplos de como utilizar caracteres especiales**
  - ls ***ue***
  - ls nuev**?**
  - ls nuevo**[**1-5**]**
  - ls –Sla **?**nu***
  - ls -lSha **[a-e]?**to***
  - ls –lSRha **[r-t]***

Búsqueda de archivos

- **Comando find:** Este comando nos permite realizar la búsqueda de ficheros dentro de nuestro sistema.
  - Su sintaxis básica es la siguiente:

find path -name nombreArchivo (el nombre entre comillas)

Las búsquedas que se realizan con el comando find se hacen de forma recursiva.

Por ejemplo para encontrar el archivo "prueba" dentro del directorio actual en el que nos encontramos habría que poner lo siguiente:

find . -name "prueba"

Si quisieramos que en la busqueda no se haga diferencia entre mayúsculas y minusculas seria de la siguiente manera

find . -iname "prueba"

Búsqueda de archivos

- El comando **find** tiene una gran cantidad de opciones, vamos ha enumerar y a explicar las principales:
  - **-type**: especifica que tipo de fichero se quiere buscar.
    - **d** - directorios
    - **f** - ficheros
    - **l** - enlaces simbólicos
    - **s** - sockets
  - **-perm**: se busca ficheros que posean los permisos que se indican. Si se pone delante de los permisos **"-"** se busca aquellos que no posean esos permisos.
  - **-user**: se buscan ficheros cuyo propietario sea el usuario indicado.
  - **-group**: se buscan ficheros que sean miembro de determinado grupo.

Búsqueda de archivos

- **-nogroup:** se buscan ficheros que sean miembro de determinado grupo.
- **-size**: busca ficheros de un determinado tamaño.
- **-newer**: busca ficheros que sean más nuevos que el **fichero 1** pero más antiguos que el **fichero 2**.
- **-mtime**: busca ficheros que han sido modificados antes de x días. Si pone el carácter "-" antes de los días se buscan ficheros en ese mismo día o antes que han sido modificados.
  - Si se pusiese "-mtime -5" se buscarían aquellos ficheros que han sido modificados hace 5 días. Si se pone el carácter "+" se buscan ficheros que no han sido modificados en los días pasados como parámetros.

Búsqueda de archivos

Ejemplos:

Buscar todos los archivos que tengan en su nombre mozilla y que se encuentren en el directorio usr.

 # find /usr -name mozilla

Buscar todas las librerías del sistema con extensión so

 # find /usr -name '*.so‘

Buscar todos los ficheros cuyo nombre contenga “gtk”

 # find / -name '*gtk*'

Buscar todos los ficheros de tamaño >= 3000 KB

 # find / -size +3000k

Búsqueda de archivos

Ejemplos:

Buscar todos los archivos que tengan en su nombre mozilla y que se encuentren en el directorio usr.

 # find /usr -name mozilla

Buscar todas las librerías del sistema con extension so

 # find /usr -name '*.so'

Buscar todos los ficheros cuyo nombre contenga "gtk"

 # find / -name '*gtk*'

Buscar archivos cuyo nombre sea "abc" sin tener en cuentas mayúsculas ni minúsculas:

 # find . -iname "abc"

Búsqueda de archivos

Ejemplos:

Buscar archivos que terminen en "wd" en el directorio /etc, sin importar las mayúsculas y minúsculas y redirigir la salida al fichero find.log.

 # find /etc -iname "*wd" > find.log

Realizar lo mismo de lo anterior solo que mandando los mensajes de error al fichero findErrors.log.

 # find /etc -iname "*wd" 2>findErrors.log

La misma búsqueda que la anterior haciendo que no se muestren los mensajes de error por la pantalla:

 # find /etc/ -iname "*wd" 2>/dev/null

Búsqueda de archivos

Ejemplos:

Buscar en el directorio actual los directorios que hay:

$ find . -type d

Buscar en el directorio actual los directorios que empiezan por A:

$ find . -type d -name "A*"

Imaginemos que quiero listar los directorios que hay en el directorio actual:

$ find ./ -maxdepth 1 -type d

Ahora imaginemos que quiero listar los ficheros que se han modificado hoy en el directorio actual:

$ find ./ -mtime 0 -type f

Búsqueda de archivos

Ejemplos busquedas complejas:

Buscar en el directorio /etc el archivo "passwd" y buscar dentro si hay un usuario llamado root:

 # find /etc -name "passwd" | xargs grep "root"

Hacer la misma búsqueda evitando que se muestren mensajes de error por pantalla.

 # find /etc -name "passwd" 2>/dev/null | xargs grep "root"

Queremos borrar todos los directorios del sistema que contengan la palabra nuevo:

 # find / -name "*nuevo*" -type d -exec rm -fr {} \; 2>/dev/null

Búsqueda de archivos

Ejemplos búsquedas complejas:

Si quisieramos borrar todos los subdirectorios del directorio /var/backup que tengan una antigüedad mayor de 20 días:

$ find /var/backup -mtime +20 -type d -exec rm -f {} \;

borrar todos los ficheros del sistema que contengan la palabra nuevo, no tenemos más que cambiar el tipo en el comando anterior:

 # find / -name "*nuevo*" -type f -exec rm -fr {} \; 2>/dev/null

queremos recopilar todos los ficheros mp3 que tenemos repartidos en diferentes directorios y moverlos a un único directorio:

 # find / -name "*.mp3" -exec mv {} /compartido/musica/ \;

Búsqueda de archivos

Tarea de búsquedas complejas:

Buscar todos los archivos con extension mp3 en el directorio home luego eliminarlos.

Buscar en el directorio /etc el archivo ‘passwd’ luego verificar que contenga la palabra root evitando que se muestren mensajes de error por pantalla.

Hacer la misma búsqueda que en el caso anterior pero esta vez que se muestren todos los usuarios del archivo passwd excepto "root":

Borrar los archivos que empiecen por "A" omitiendo mayúsculas y minúsculas en el directorio actual y hacia abajo:

Lo mismo que en el caso anterior pero que se pida confirmación antes de borrar.

Búsqueda de archivos

locate: Es un comando de búsqueda de archivos, bastante parecido al comando **find**. La diferencia de **locate** es que la búsqueda la hace en una base de datos indexada para aumentar significativamente la velocidad de respuesta.

$ **locate archivo**

grep: Es un comando que permite buscar, dentro de los archivos, las líneas que concuerdan con un patrón, tiene varias opciones.

-c En lugar de imprimir las líneas que coinciden, muestra el número de líneas que coinciden.

-e PATRON nos permite especificar varios patrones de búsqueda o proteger aquellos patrones de búsqueda que comienzan con el signo **-** .

Búsqueda de archivos

-r busca recursivamente dentro de todos los subdirectorios del directorio actual.

-v nos muestra las líneas que no coinciden con el patrón buscado.

-i ignora la distinción entre mayúsculas y minúsculas.

-n Numera las líneas en la salida.

-E nos permite usar expresiones regulares. Equivalente a usar **egrep**.

-o le indica a grep que nos muestre sólo la parte de la línea que coincide con el patrón.

-f ARCHIVO extrae los patrones del archivo que especifiquemos. Los patrones del archivo deben ir uno por línea.

-H nos imprime el nombre del archivo con cada coincidencia**.**

Búsqueda de archivos

Ejemplos:

Buscar todas las líneas que contengan palabras que comiencen por **a** en un archivo:

$ grep '\<a.*\>' archivo

Otra forma de buscar, sería:

$ cat archivo | grep "\<a.*\>"

Mostrar por pantalla, las líneas que contienen comentarios en el archivo /boot/grub/menu.lst:

$ grep "#" /boot/grub/menu.lst

Enviar a un fichero las líneas del archivo /boot/grub/menu.lst que no son comentarios:

$ grep -v "#" /boot/grub/menu.ls

Búsqueda de archivos

Ejemplos:

Contar el número de interfaces de red que tenemos definidos en el fichero /etc/network/interfaces:

$ grep -c “iface” /etc/network/interfaces

Mostrar las líneas de un fichero que contienen la palabra root o bueno:

$ grep -e “root” -e “bueno” nombre_archivo

Mostrar las líneas de un fichero que contienen la palabra root o bueno, numerando las líneas de salida:

$ grep -n -e “root” -e “bueno” nombre_archivo

Mostrar los ficheros que contienen la palabra TOLEDO en el directorio actual y todos sus subdirectorios:

$ grep -r "TOLEDO" *

Búsqueda de archivos

Ejemplos 1 con expresiones regulares:

Obtener la dirección MAC de la interfaz de red eth0 de nuestra máquina:

$ ifconfig eth0 | grep -oiE '([0-9A-F]{2}:){5}[0-9A-F]{2}'

Sacamos la dirección MAC de la interfaz eth0 de nuestra máquina haciendo un: **ifconfig eth0** y aplicando el filtro grep:

grep -oiE '([0-9A-F]{2}:){5}[0-9A-F]{2}'

Las opciones que he usado en grep son:

-o Indica que la salida del comando debe contener sólo el texto que coincide con el patrón, en lugar de toda la línea.

-i Ignora la distinción entre mayúsculas y minúsculas.

-E Indica que vamos a usar una expresión regular extendida.

Búsqueda de archivos

Ejemplos 1 con expresiones regulares:

En cuanto a la expresión regular, podemos dividirla en dos partes: **([0-9A-F]{2}:){5}** Buscamos 5 conjuntos de 2 caracteres seguidos de dos puntos.

[0-9A-F]{2} seguido por un conjunto de dos caracteres. Como las direcciones MAC se representan en hexadecimal, los caracteres que buscamos son los números del 0 al 9 y las letras desde la A a la F.

Búsqueda de archivos

Ejemplos 2 con expresiones regulares:

Extraer la lista de direcciones de correo electrónico de un archivo:

#grep -Eio '[a-z0-9._-]+@[a-z0-9.-]+[a-z]{2,4}' fichero.txt

Utilizo las mismas opciones que en el caso anterior:
-o Indica que la salida del comando debe contener sólo el texto que coincide con el patrón, en lugar de toda la línea, como es lo habitual.
-i Lo he usado para que ignore la distinción entre mayúsculas y minúsculas.

-E Indica que vamos a usar una expresión regular extendida.
Analicemos ahora la expresión regular:

[a-z0-9._-]+@[a-z0-9.-]+[a-z]{2,4}

Búsqueda de archivos

Ejemplos 2 con expresiones regulares:

Al igual que antes, la vamos dividiendo en partes:

[a-z0-9._-]+ Una combinación de letras, números, y/o los símbolos . _ y - de uno o más caracteres **@** seguido de una arroba **[a-z0-9.-]+** seguido de una cadena de letras, números y/o los símbolos . y –**[a-z]{2,4}** seguido de una cadena de entre dos y cuatro caracteres.

Tarea de búsquedas con grep

Obtener la dirección IP de la interfaz de red eth0 de nuestra máquina:

Comando Alias en Linux

alias: Un alias es un nombre mas entendible que se le da a un comando.

- El objetivo de crear alias de comandos, de forma que sea más sencillo de recordar para nosotros.

  #alias nombre_entendible=comando

Ejemplos

#alias apagar=halt

#alias listar_archivos=ls

#alias leer_archivo=cat

 # alias listar_archivos='ls -l'

unalias: Permite eliminar un alias personalizado.

 # unalias listar_archivos

Manejo de Procesos (Dem)

Comando ps: El comando ps es el que permite informar sobre el estado de los procesos, esta basado en el sistema de archivos /proc, es decir, lee directamente la información de los archivos que se encuentran en este directorio.

Tiene una gran cantidad de opciones, incluso estas opciones varían dependiendo del estilo en que se use el comando.

p o PID	Process ID, número único o de identificación del proceso.
P o PPID	Parent Process ID, padre del proceso
U o UID	User ID, usuario propietario del proceso
t o TT o TTY	Terminal asociada al proceso, si no hay terminal aparece entonces un '?'

Manejo de Procesos (Dem)

T o TIME	Tiempo de uso de cpu acumulado por el proceso
c o CMD	El nombre del programa o comando que inició el proceso
RSS	Resident Sise, tamaño de la parte residente en memoria en kilobytes
SZ o SIZE	Tamaño virtual de la imagen del proceso
NI	Nice, valor nice (prioridad) del proceso, un número positivo significa menos tiempo de procesador y negativo más tiempo (-19 a 19)
C o PCPU	Porcentaje de cpu utilizado por el proceso

Manejo de Procesos (Dem)

STIME	Starting Time, hora de inicio del proceso
S o STAT	•Status del proceso, estos pueden ser los siguientes R runnable, en ejecución, corriendo o ejecutándose •S sleeping, proceso en ejecución pero sin actividad por el momento, o esperando por algún evento para continuar •T sTopped, proceso detenido totalmente, pero puede ser reiniciado •Z zombie, difunto, proceso que por alguna razón no terminó de manera correcta, no debe haber procesos zombies •D uninterruptible sleep, son procesos generalmente asociados a acciones de IO del sistema •X dead, muerto, proceso terminado pero que sigue apareciendo, igual que los Z no deberían verse nunca

Manejo de Procesos (Dem)

- **Opciones de administración del comando ps:**
  - **-e** mostrar todos los procesos
  - **-d** mostrar todos excepto administradores de sesión
  - **-f** mostrar todas las columnas de manera amplia
  - **-a** mostrar todos incluyendo procesos de otros usuarios
  - **-u** mostrar por identificador de usuario
  - **-x** mostrar procesos con control de las ttys
  - **-r** mostrar solo procesos que están corriendo
  - **-v** mostrar memoria virtual
  - **-H** muestra la jeraquia de procesos

Manejo de Procesos (Dem)

Jobs : Los jobs son procesos que se ejecutan en segundo plano, para poder mandar un proceso a segundo plano background con el objeto de liberar la terminal y continuar trabajando se utiliza el operador '&' .

para ver los procesos actuales en ejecución tenemos el comando **jobs.**

```
#> yes > /dev/null &
[1] 26837
#> ls -laR > archivos.txt &
[2] 26854
#> jobs
[1]-  Running                 yes >/dev/null &
[2]+  Running                 ls --color=tty -laR / >archivos.txt &
#> jobs
[1]-  Stopped                 yes >/dev/null &
[2]+  Stopped                ls --color=tty -laR / >archivos.txt &
#> fg %1
#> jobs
[1]+  Running                 yes >/dev/null &
[2]-  Stopped                ls --color=tty -laR / >archivos.txt &
```

Manejo de Procesos (Dem)

- **Ejemplos comando ps:**

```
>#  ps -e     (-e muestra todos los procesos)
  PID TTY          TIME CMD
    1 ?        00:00:01 init
    2 ?        00:00:00 kthreadd
    3 ?        00:00:00 migration/0
    4 ?        00:00:00 ksoftirqd/0

#> ps -ef     (-f muestra opciones completas)
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 10:12 ?        00:00:01 init [5]
root         2     0  0 10:12 ?        00:00:00 [kthreadd]
...
root      6130  5662  0 10:24 pts/0    00:00:00 su -
root      6134  6130  0 10:24 pts/0    00:00:00 -bash
sergon    6343  5604  0 10:28 ?        00:00:00 kio_file [kdeinit] file /home/sergon/tmp/k
root      6475  6134  0 10:38 pts/0    00:00:00 ps -ef

#> ps -eF     (-F muestra opciones completas extra)
UID        PID  PPID  C    SZ   RSS PSR STIME TTY          TIME CMD
root         1     0  0   412   556   1 16:59 ?        00:00:01 init [5]
root         2     0  0     0     0   1 16:59 ?        00:00:00 [kthreadd]
sergon    8326  8321  0   902  1272   0 17:07 ?        00:00:00 /bin/sh /usr/lib/firefox-2
sergon    8331  8326  4 53856 62604   0 17:07 ?        00:00:50 /usr/lib/firefox-2.0.0.8/m
sergon    8570  7726  2 15211 37948   0 17:17 ?        00:00:10 quanta
```

Manejo de Procesos (Dem)

- **Ejemplos comando ps:**

```
#> ps ax      (formato BSD sin guión, a muestra todos, x sin mostrar tty)
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss     0:01 init [5]
    2 ?        S<     0:00 [kthreadd]
    3 ?        S<     0:00 [migration/0]
    4 ?        S<     0:00 [ksoftirqd/0]

#> ps aux    (formato BSD sin guión, u muestra usuarios y demás columnas)
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0   1648   556 ?        Ss   16:59   0:01 init [5]
root         2  0.0  0.0      0     0 ?        S<   16:59   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S<   16:59   0:00 [migration/0]
root         4  0.0  0.0      0     0 ?        S<   16:59   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   16:59   0:00 [migration/1]

#> ps -eo user,pid,tty    (-o output personalizado, se indican los campos separados por coma,
USER       PID TT
root         1 ?
root         2 ?
sergon    8570 tty 1
root      8876 pts/1

#> ps -eH  (muestra árbol de procesos)
#> ps axf  (lo mismo en formato BSD)
#> ps -ec  (el comando  que se esta ejecutando, sin la ruta, solo el nombre real)
#> ps -el  (muestra formato largo de varias columnas, muy práctico)
#> ps L    (No muestra procesos, lista todos los códigos de formatos)
```

Manejo de Procesos (Dem)

Comando pstree: Muestra los procesos en forma de árbol

Opciones

-A y –G para que te un árbol con líneas con líneas estilo ASCII

-u para que te muestre entre paréntesis el usuario propietario del proceso.

```
#> pstree -AGu
init---acpid
     |-atd(daemon)
     |-automount----2*[{automount}]
     |-avahi-daemon(avahi)
     |-beagled(sergon)----7*[{beagled}]
     |-beagled-helper(sergio)----3*[{beagled-helper}]
     |-compiz(sergon)----kde-window-deco
     |-console-kit-dae----61*[{console-kit-dae}]
     |-crond
     |-dbus-daemon(messagebus)
     |-dbus-daemon(sergio)
     |-dbus-launch(sergio)
     |-dcopserver(sergio)
     |-dhclient
     |-gam_server(sergio)
     |-gconfd-2(sergio)
     |-hald(haldaemon)----hald-runner(root)----hald-addon-acpi(haldaemon)
     |                                        |-hald-addon-cpuf
     |                                        |-hald-addon-inpu
     |                                        |-hald-addon-stor
```

Manejo de Procesos (Dem)

Comando kill: El comando kill, que literalmente quiere decir matar, sirve no solo para matar o terminar procesos sino principalmente para enviar señales (signals) a los procesos

```
#> kill -l     (lista todas las posibles señales que pueden enviarse a un proceso)
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL
 5) SIGTRAP      6) SIGABRT      7) SIGBUS       8) SIGFPE
 9) SIGKILL     10) SIGUSR1     11) SIGSEGV     12) SIGUSR2
13) SIGPIPE     14) SIGALRM     15) SIGTERM     16) SIGSTKFLT
17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU
25) SIGXFSZ     26) SIGVTALRM   27) SIGPROF     28) SIGWINCH
29) SIGIO       30) SIGPWR      31) SIGSYS      34) SIGRTMIN
35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3  38) SIGRTMIN+4
39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12
47) SIGRTMIN+13 48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14
51) SIGRTMAX-13 52) SIGRTMAX-12 53) SIGRTMAX-11 54) SIGRTMAX-10
55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7  58) SIGRTMAX-6
59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
```

Manejo de Procesos (Dem)

Ejemplos con el comando kill:

#kill -9 [id proceso] mata el proceso sin previo aviso

#kill -15 [id proceso] le da un aviso para que el proceso se cierre

```
#> kill -9 11428          (termina, mata un proceso completamente)
#> kill -SIGKILL 11428    (Lo mismo que lo anterior)
```

Comando killall: Este comando funciona de manera similar a kill, pero con la diferencia de en vez de indicar un PID se indica el nombre del programa.

```
#> killall -l
#> killall -HUP httpd
#> killall -KILL -i squid
```

-l permite listar procesos a matar

-HUP permite dar señal reinicio al proceso

-i permite pedir confirmacion

Manejo de Procesos (Dem)

Comando pkill: Nos permite matar un proceso por su nombre

#pkill -9 [nombre_proceso]

Comando nice: Permite cambiar la prioridad de un proceso.

Con nice es posible iniciar un programa (proceso) con la prioridad modificada, más alta o más baja según se requiera.

Las prioridades van de -20 (la más alta) a 19 (la más baja), solo root o el superusuario puede establecer prioridades negativas que son más altas.

Con la opción -l de ps es posible observar la columna NI que muestra este valor.

```
#> nice                (sin argumentos, devuelve la prioridad por defecto )
0
#> nice -n -5 comando   (inicia comando con una prioridad de -5, lo que le da más tiempo de cpu)
```

Manejo de Procesos (Dem)

Comando renice: Este comando permite alterarla en tiempo real, sin necesidad de detener el proceso.

Ejemplos:

```
#> nice -n -5 yes    (se ejecuta el programa 'yes' con prioridad -5)
                     (dejar ejecutando 'yes' y en otra terminal se analiza con 'ps')
#> ps -el
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0 12826 12208  4  75  -5 -   708 write_ pts/2    00:00:00 yes

#> renice 7 12826
12826: prioridad antigua -5, nueva prioridad 7
#> ps -el
F S   UID   PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S     0 12826 12208  4  87   7 -   708 write_ pts/2    00:00:15 yes
```

- (obsérvese el campo NI en el primer caso en -5, y en el segundo con renice quedó en 7, en tiempo real)

Manejo de Procesos (Dem)

El sistema es multitarea, podemos dejar múltiples procesos corriendo en background.

- Con **control+z** llevamos el siguiente proceso a (segundo plano).

bg: nos permite listar el ultimo proceso enviado a segundo plano

#bg

fg: permite colocar un proceso a primer plano (foreground).

#fg [numero de proceso]