U4 Administración de Sistemas Operativos Linux

Parte V. Scripts

Implantación de Sistemas Operativos

grep, egrep y grep -E

- Permite buscar las líneas que contengan un patrón o expresión regular en un texto (grep ‘expresión’):
  - **-i** no distingue mayúsculas de minúsculas
  - **-w** para que la palabra deba ser completa para que considere que coincide con el patrón
  - **-c** cuenta el número de líneas
  - **-v** busca la que **NO** contenga el patrón
- Si lo usamos con expresiones regulares usaremos **egrep** o **grep –E.** Entre otras cosas permiten el uso de caracteres:
  - **egrep ‘^f’** buscará palabras que empiecen por f
  - **egrep ‘in$’** buscará palabras que terminen por in
  - **egrep ‘...’** buscará palabras con 3 caracteres (el punto sustituye a un carácter)

grep, egrep y grep -E

- Ejemplo: obtén los usuarios con shell bash del sistema

```
grep "/bin/bash" /etc/passwd | cut -d':' -f1
```

- Ejemplo: obtén el número de usuarios con shell bash del sistema

```
grep –c "/bin/bash" /etc/passwd | cut -d':' -f1
```

- Ejemplo: buscar líneas con un dni (usamos egrep para expresiones regulares extendidas).

```
cat fichero.txt | egrep -i -w '[0-9]{8}[A-Z]'
```

sed

- El comando **sed** tiene varias utilidades:
  - ***Borrar líneas*** de un fichero (o de la salida de un comando pasada por tubería).
    - Indicando el número de línea seguida de la letra 'd', la borra → **sed '3d' fichero**
    - Ejemplo: Eliminar la 1ª línea → **ls –l | sed '1d'**
  - ***Imprimir líneas*** de un fichero (o de la salida de un comando pasada por tubería).
    - -n y el número de línea seguido de 'p', imprime sólo esa línea → **sed -n '3p' fichero**. (-n para que no imprima todo el fichero, solo las coincidencias, p para imprimir por pantalla el patrón).
    - Podemos indicar un rango de líneas separadas por coma. El segundo carácter puede ser '$' que indica hasta el final → **sed '4,$p' fichero** (imprime desde la 4ª línea hasta el final).

sed

- Podemos concatenar instrucciones con la opción **-e** delante de cada → **sed -n -e '3p' -e '9p'** (Imprime las líneas 3 y 9)
- Podemos indicar que nos imprima sólo las líneas pares, impares, cada 3, etc. Simplemente indicando **INICIO~SALTO**. Por ejemplo, mostrar las líneas impares es empezar por la primera mostrando cada 2 líneas → **sed -n '1~2p'**
- Para recorrer un fichero línea por línea lo podemos hacer con **sed '$xp'** donde x será el número de línea.

- ***Sustituir cadenas***. La sintaxis de sed para sustituir la cadena1 por la cadena2 es (las expresiones tienen que ir entre barras):

  sed 's/cadena1/cadena2/g'

  - Opciones:
    - **s:** antes de la expresión, indica sustitución (1 vez)
    - **g**: reemplaza todas las ocurrencias de la línea. Por defecto solo reemplaza la primera.

sed

- ***Sustituir cadenas***. La sintaxis de sed para sustituir la cadena1 por la cadena2 es (las expresiones tienen que ir entre barras, para indicar que buscamos un patron):

  sed 's/cadena1/cadena2/g'

  - Opciones:
    - **s:** antes de la expresión, indica sustitución (solo 1ª coincidencia)
    - **g**: reemplaza todas las ocurrencias de la línea. Por defecto solo reemplaza la primera.
    - **I** (i mayúscula): no diferencia entre mayúsculas/minúsculas.
    - **numero:** reemplaza la coincidencia el número indicado de veces (el resto no).

      echo 123123123 | sed 's/123/abc/1' → abc123123

sed

- Ejemplo: Tenemos un fichero cuyas columnas están separadas, bien por coma, o por punto y coma, y necesitamos un separador común para poder separarlas con el comando **cut**. Vamos a sustituir los punto y coma por comas y mostrar la segunda y cuarta columnas
    - **cat fichero.txt | sed -r 's/;/,/g' | cut -d “,” -f 2,4**
- Reemplazar cadena sólo en las líneas que tengan esa cadena:
    - **sed '/cadena_a_buscar/ s/vieja/nueva/g' fichero > fichero2**
- Reemplazar múltiples cadenas (A o B):
    - **sed 's/cadenasrc1\|cadenasrc2/cadena_nueva/g’**
- Si queremos borrar una coincidencia, el segundo patrón irá en blanco:
    - **sed '/cadena_a_buscar/ s/vieja/ /g' fichero > fichero2**
- “&” hace referencia a lo localizado en el patrón 1 y añadimos algo al patrón encontrado:

```
marina@marina-VirtualBox:~$ echo "asgiahngifadgih" | sed s'/[a-z]/&:/'g
a:s:g:i:a:h:n:g:i:f:a:d:g:i:h:
```

- Tanto para borrar e imprimir líneas, como para sustitución de cadenas se pueden emplear expresiones regulares.

Sed y expresiones regulares

- Añadimos la opción **-r**, para trabajar con expresiones regulares.
- **sed -r '/expresión/d'** → Mostraría todas las líneas excepto las que tengan coincidencia con la expresión, que las elimina (d → borra).
- **sed -n -r '/expresión/p'** → Mostraría las líneas que tengan alguna coincidencia con la expresión. Equivalente al comando grep -E 'expresión' o egrep ‘expresión’. La opción p imprime las líneas que coinciden con la expresión. Se usa con la opción -n, ya que si no, lo imprime todo, y no sólo lo que coincide: 'expresión’.
- **sed -r 's/casa/ mansión/g'** → Sustituye la palabra casa por la palabra mansión. La opción g al final indica que se sustituya todas las veces que aparezca (por defecto sólo se sustituye la primera).
- Podemos usar las mismas opciones descritas anteriormente: s, g, I, número.

Sed y expresiones regulares

- Para modificar el fichero, tenemos que usar la opción **i.** Es conveniente hacer una copia del fichero, antes de modificarlo.

- Ejemplos:
  - **sed -r 's/^hola/Adiós/i'** → Sólo se sustituye hola por adiós si es la primera palabra en el texto.
  - **sed -r 's/^error$/x_x/i'** → Sólo las líneas cuyo único texto sea 'error' se sustituyen por 'x_x'
  - '/a[0-9]*/' → **a a6 a908345**
  - '/a[0-9]+/' → ~~**a**~~ **a6 a908345**
  - '/a[0-9]?bc/' → **abc a6bc** ~~**a908bc**~~
  - '/a[0-9]{3}bc/' → ~~**a6bc**~~ **a456bc**
  - '/a[0-9]{2,4}bc/' → ~~**a67867bc**~~ **a45bc a123bc**
  - '/a[0-9]{3,}bc/' → **a67867bc** ~~**a45bc**~~ **a123bc**
  - '/(a[0-9])+bc/' → **a4a5bc** ~~**a1a23bc**~~ **a0bc**
  - '/(ab|cd)+[0-9]/' → ~~**acbd3**~~ **ababcdab3** ~~**abc3**~~

awk

- Es un comando más complejo y potente que puede hacer lo mismo y más que cut, grep y sed juntos. En este caso, sólo vamos a tratar con su capacidad de mostrar columnas de forma desordenada (cut -f 3,1 no muestra primero la 3ª y luego la 1ª).

- El separador se indica con la opción -F “separador” (igual que -d en cut pero puede ser más de un carácter). Luego indicamos cómo queremos que aparezcan las columnas de esta forma:
  - **awk -F ';' '{print $2" va antes que "$1}’**

```
marina@marina-VirtualBox:~$ cat /etc/passwd | cut -d ':' -f 3,1
root:0
daemon:1
bin:2
```

```
marina@marina-VirtualBox:~$ cat /etc/passwd | awk -F ':' '{ print $3":"$1 }'
0:root
1:daemon
2:bin
```

awk

- El comando al contrario que cut, **SÍ** que elimina el separador. Si lo queremos volver a poner (o cualquier texto por en medio), debemos indicarlo como texto normal. El texto normal se encierra entre comillas dobles “”, también podemos separar por comas, pero en este caso, el comando además introducirá un espacio de separación automáticamente.
  - **awk -F ';' '{print $2" va antes que "$1}' <==> awk -F ';' '{print $2,"va antes que", $1}'**

```
marina@marina-VirtualBox:~$ cat /etc/passwd | awk -F ':' '{ print $3,":",$1 }'
0 : root
1 : daemon
2 : bin
```

awk

- En resumen $NÚMERO hace referencia a la columna a imprimir, mientras que el texto que queramos introducir entre medias, lo debemos poner encerrado entre comillas.
- El comando awk también admite operaciones matemáticas con las columnas:
  - **awk -F ';' '{print "Precio: "$1", Cantidad: "$2“,, Total: "$1*$2}’**
  - **cat /etc/passwd | awk -F ':' '{ print $3,":",$4,"="$3+$4}'**
  - **'{print toupper($1), tolower($2), $3}'** → Funciones predefinidas para transformar a mayúsculas y minúsculas.
  - **cat /etc/passwd | awk -F ':' '{ print toupper($1),":",toupper($7)}'**

Depuración

- La depuración o debugging de scripts es el proceso de identificar y corregir errores de programación.
- Hay 2 formas:
  - -v: Muestra cada línea completa del script antes de ser ejecutada
  - -x: Muestra cada línea abreviada del script antes de ser ejecutada
- Podemos aplicar ambos métodos modificando la primera línea del script o cambiando las opciones de ejecución de la Shell
- sh –v o sh –x + nombre script.
- **bash –v o bash –x + nombre script.**