Tareas de explotación de Sistemas de Archivos

Sumario

Mediciones del Sistema de Archivos

En GNU/Linux todo recurso con el que interactúa el sistema operativo se representa como un archivo. De este modo disponemos de un interfaz uniforme de acceso a los recursos, independientemente de su tipo.

En esta práctica vamos a ver el uso de algunas herramientas útiles para medir el uso de recursos relacionados con los sistemas de archivos.

Medición de uso de espacio ocupado

El comando du (disk use) permite conocer el espacio ocupado en un sistema de archivos por archivos o directorios. Su sintaxis es

du [opciones] <ruta_directorio_o_archivo>

Mediante el comando anterior podemos medir el espacio utilizado por archivos dentro de un directorio

Obtener el tamaño de todos los archivos dentro de un directorio

Podemos obtener el tamaño de todos los archivos dentro de un directorio dado

Por ejemplo en el directorio home

```
cd
du $(pwd)
```

El comando mostraría el tamaño de todos los archivos y directorios dentro del directorio home del usuario. El tamaño se mostraría en bytes por defecto. Si queremos ver el tamaño en unidades más fáciles de interpretar, usaríamos la opción -h (human)

```
du -h $(pwd)
```

Cualquier de los dos comandos anteriores muestra un listado de archivos con su correspondiente tamaño, pero, ¿y si quiero ver el tamaño total del directorio?

Entonces usamos la opción **-s** (summary)

```
du -hs $(pwd)
```

El resultado sería una única línea para el directorio, en la que se muestra el tamaño total de sus contenidos.

NOTAS

- Por lo general, al trabajar con comandos de línea de comandos, existe una opción que muestra la ayuda textual de la herramienta correspondiente. En este caso, y suele ser algo habitual, la opción es ?help, o abreviadamente, -h

```
du --help
```

Cuando utilicemos du es importante que el usuario que lo invoque tenga permisos de lectura en los directorios y archivos contenidos dentro del directorio a procesar. De lo contrario obtendremos errores de permiso denegado.

Medición de uso de espacio libre

Obtener el tamaño de espacio libre en sistemas de archivos

El comando df (disk free), nos permite obtener el espacio libre dentro de los sistemas de archivos. Su sintaxis es

df [opciones] [fichero]

Vemos que tanto las opciones como el fichero son opcionales. Si invocamos df sin opciones ni ficheros obtenemos un listado del espacio libre de todos los sistemas de archivos montados

```
df
```

La salida de este comando nos mostrará el espacio usado y disponible en todos los sistemas de archivos, expresando las unidades de espacio libre y usado en **Kilobytes**.

Para visualizar el espacio libre y ocupado en unidades de medida binarias más fáciles de visualizar, podemos utilizar la opción **-h** (expresado en unidades potencia de 2, KiB, MiB, GiB), o bien -H (expresando las unidades en potencias de 10, KB, MB, GB?)

```
df -h /
```

Mostraría el espacio usado y libre del sistema de archivos en el que reside el directorio raíz

NOTA

Como siempre la opción --help del comando df nos permite ver la ayuda textual del mismo

Buscar archivos y directorios según metadatos

Mediante el comando **find** podemos localizar archivos y directorios que verifiquen determinadas condiciones en sus metadatos.

Los metadatos son aquellos atributos asociados a archivos y directorios almacenados en los i-nodos.

find [opciones] [path_busqueda] [expresión]

NOTA

Como siempre la opción **--help** del comando df nos permite ver la ayuda textual del mismo

El parámetro **path_busqueda**, representa a partir de que punto del sistema operativo se realiza la búsqueda. Sino se especifica la búsqueda comienza en el directorio actual

La **expresión** indica el elemento utilizado para definir la condición que deben de verificar los elementos del sistema de archivos a localizar.

Veamos un ejemplo

```
find $(pwd) -name I*
```

Buscaría en el directorio actual todos los archivos y directorios cuyo nombre empiece con la letra I

```
find $(pwd) -type d
```

Buscaría en el directorio actual todos los elementos de tipo directorio

Para ver una lista completa de todas las opciones de construcción de expresiones de búsqueda, ademas de --help podemos consultar las páginas del manual

```
man find
```

Vamos a localizar aquellos archivos, en el directorio actual, de 5MB o más de tamaño que hayan sido accedidos dentro del directorio actual hace más de 5 días

```
find $(pwd) -atime +5 -size +5M
```

Especificando el valor numérico en negativo (-) indicaremos que el tiempo de acceso debe de ser inferior al número de días indicado, por ejemplo:

```
find $(pwd) -atime -5 -size +5M
```

En este caso el comando busca aquellos archivos de 5MB o más de tamaño que hayan sido accedidos en los últimos 5 días

NOTA

El valor numérico que acompaña a algunas opciones, como en caso del -atime o -size anteriores, tiene la siguiente interpretación:

- N: Busca elementos que cumplan exactamente el valor numérico indicado
- +N: Busca elementos que excedan el valor indicado
- -N: Busca elementos que estén por debajo del valor numérico indicado

El siguiente comando localiza, en el directorio actual, archivos regulares que hayan sido modificados en las últimas 24 horas, y que pertenezcan al usuario root

```
find $(pwd) -mtime 0 -type f -user root
```

Un aspecto interesante de find es que permite ejecutar acciones sobre los elementos que verifiquen la búsqueda, por ejemplo borrarlos. Podemos utilizar **-exec** por ejemplo

```
find $(pwd) -maxdepth 2 -type f -exec file {} \;
```

El comando anterior se ejecuta con la opción **maxdepth**, es una opción, no un parámetro, ojo. Esta opción limita el número de niveles que se ?descienden? en la búsqueda. De este modo acotamos la dimensión de la prueba. En este caso buscamos dos niveles por debajo del directorio actual.

La opción **type** indica que solamente nos interesan archivos regulares. El parámetro **-exec** especifica una acción, en este caso la ejecución de un **comando file** para cada uno de los elementos que verifican la condición del find.

El elemento en la sintaxis **{}** es sustituído por cada elemento que verifica la condición, es una forma genérica de expresar la ruta del archivo sin especificar el mismo. Por último el comando debe de ser terminado con un ?**;**?, como éste es un carácter significativo e interpretable por la shell, lo escapamos con ?\?.

Localizar los archivos usados por un proceso

En ocasiones es importante conocer que archivos y directorios están siendo utilizados por un proceso determinado, o bien saber que archivos y directorios de un directorio o dispositivo concreto están abiertos, incluso puede ser de interés el conocer los servicios o puertos que están a la escucha.

Para resolver estas, y otras, importantes cuestiones podemos utilizar el comando **lsof** (list open files).

Ver los archivos abiertos por un determinado proceso

Con la opción -p podemos indicar el PID (proccess ID) del proceso del cual queremos saber que elementos del sistema de archivos está utilizando

```
lsof -p 1000
```

El comando anterior mostraría los archivos abiertos por el proceso con PID 1000. Es evidente que, para poder utilizar este comando, necesitamos conocer el PID del proceso en cuestión. Si solamente sabemos el nombre del proceso, podremos apoyarnos en el comando pidof, para obtener el PID del proceso a partir de su nombre.

Por ejemplo vamos a ver los archivos abiertos por todos los procesos firefox-esr en ejecución en el sistema

```
for pid in $(pidof firefox-esr); do lsof -p $pid;done | more
```

El comando anterior ejecuta un **bucle for** (ciclo) consistente en ejecutar el comando **lsof -p** para cada uno de los **PID** de procesos asociados al ejecutable **firefox-esr**, obtenidos mediante la expresión **$(pidof firefox-esr)**. La salida del ciclo se envía al paginador more para poder verla pantalla a pantalla.

Ver los archivos abiertos para una determinada ruta

```
lsof /
```

Mostraría los archivos y directorios abiertos dentro del directorio raíz

Ver los archivos abiertos por un determinado usuario

También es posible ver los archivos y directorios abiertos asociados a un usuario, para ello usamos la opción -u

```
lsof -u $(whoami)
```

Ver los puertos de servicios en escucha

Otro uso bastante frecuente, aunque existen otras herramientas para ello, es el conocer los servicios a la escucha en la máquina, para ello usamos la opción -i

```
lsof -i
```

Si queremos hacerlo para un puerto concreto

```
lsof -i:80
```

Clonado de Sistemas de Archivos

Suponemos los dispositivos

- **/dev/loop0**: Dispositivo virtual de bloques sobre un archivo de 1000M
    - Contiene una partición que abarca todo el espacio del dispositivo
    - Contiene un sistema de archivos ext4
    - Dentro del sistema de archivos nos encontramos la siguiente estructura de archivos y directorios

```
./dir1:
archivo1

./dir2:
archivo2

./dir3:
archivo3
```

- **/dev/loop1**: Dispositivo virtual de bloques sobre un archivo de 1000M, vacío

Clonado del dispositivo loop0 en loop1 en un paso

```
dd if=/dev/loop0 of=/dev/loop1
```

Este comando es una copia bloque a bloque del primer dispositivo, loop0, en el segundo dispositivo, loop1. Al ser una copia de todos los bloques del dispositivo:

- Se copiará la tabla de particiones
- Se copiará toda la estructura del SA
- Se copiarán todos los datos del SA

Para comprobarlo vamos a montar el SA ext4 que debería de haber sido clonado, junto con todos sus datos, en el proceso de copiado bloque a bloque del dispositivo loop0 en loop1. Ejecutamos los comandos:

```
[ -d /mnt/ext4 ] || mkdir /mnt/ext4
mount /dev/loop1p1 /mnt/ext4
ls -R /mnt/ext4
```

debería mostrar la misma salida anterior:

```
./dir1:
archivo1

./dir2:
archivo2

./dir3:
archivo3
```

Clonado en dos pasos de una partición de loop0 en loop1

En primer lugar clonamos la tabla de particiones (si es necesario mantener la estructura de particiones):

```
sfdisk -d /dev/loop0 | sfdisk /dev/loop1
```

En segundo lugar clonamos los bloques de la partición correspondiente al SA

```
dd if=/dev/loop0p1 of=/dev/loop1p1
```

Notar que para que el comando anterior funcione tienen que darse varias condiciones:

- Que existan las particiones origen y destino
- Que la partición de destino tenga un tamaño igual o superior a la partición de origen

Chequeo de Sistemas de Archivos

Linux dispone de una herramienta similar a Scandisk de Windows, utilizada para detectar posibles errores en la estructura de los sistemas de archivos. Vamos a ejecutar el comando en cuestión contra la **partición 1** del dispositivo **loop0**, identificada como /**dev**/**loop0p1**

```
fsck /dev/sdc1
```

IMPORTANTE: Comprobad que da una advertencia indicando que para poder ejecutar el comando es fundamental que el sistema de archivos de esa partición esté desmontada!!!...si lo ejecutáis podéis dañarr el SA

Decimos que no a la comprobación y vamos previamente a desmontar el sistema de archivos (supongamos que está montado en /mnt/ext4):

```
umount /mnt/ext4
```

Ahora ya podemos ejecutar con seguridad fsck

```
fsck /dev/loop0p1
```

fsck soporta múltiples opciones de funcionamiento, puede verse una lista completa ejecutando

```
fsck --help
```

NOTA: Para chequear sistemas de archivos NTFS podemos utilizar la herramienta **ntfsfix**

Ejemplo:

```
ntfsfix /dev/sdc2
```

Volver

JavierFP 14:21 22 nov 2018 (CET)