Comando qemu-img

Sumario

Gestión de volúmenes con qemu-img

Ver información de una imagen o volumen

Con el comando info vemos información asociada a una imagen

```
qemu-img info win10.qcow2
```

Mostraría

```
image: win10.qcow2
file format: qcow2
virtual size: 20G (21474836480 bytes)
disk size: 9.0G
cluster_size: 65536
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

Creación de una imagen

Pueden crearse archivos de imagen al vuelo con el comando create

```
qemu-img create -f qcow2 test.qcow2 1G
```

Crearía la imagen test.qcow2 de 1GiB de tamaño en formato qcow2.

Con la **opción -b** filename (análogamente con -o backing_file=filename) podemos crear una imagen a partir de otra existente, que se pasará como argumento de esa opción, de modo que la nueva imagen solo almacenará las diferencias respecto a la imagen base.

Nota sobre el espacio ocupado por los volúmenes

Cuando se crea un volumen de almacenamiento en formato qcow2 el espacio ocupado por el mismo es el declarado en la creación del volumen, no el espacio realmente ocupado por los datos que contiene.

Veamos los espacios ocupados por los volúmenes

```
virsh vol-list default --details
```

```
guest1-deb9.qcow2   /var/lib/libvirt/images/guest1-deb9.qcow2   file  20,00 GiB    3,32 MiB
```

```
 w10_cliente.qcow2   /var/lib/libvirt/images/w10_cliente.qcow2   file  20,00 GiB    2,24 GiB
 w2016.qcow2         /var/lib/libvirt/images/w2016.qcow2         file  30,00 GiB   10,92 GiB
 w2016_server.qcow2  /var/lib/libvirt/images/w2016_server.qcow2  file  30,00 GiB    1,21 GiB
 win10.qcow2         /var/lib/libvirt/images/win10.qcow2         file  20,00 GiB    9,01 GiB
```

Si nos fijamos en la primera línea del listado veremos que hay un volumen de 20 GiB, del cual solo se están utilizando 3,32 MiB

Ahora veamos el espacio ocupado por ese volumen en el sistema de archivos

```
ls -lh /var/lib/libvirt/images
```

Mostraría:

-rw------- 1 root root 21G nov 9 17:15 guest1-deb9.qcow2

```
-rw-r--r-- 1 root         root         2,3G nov  7 19:44 w10_cliente.qcow2
-rw-r--r-- 1 libvirt-qemu libvirt-qemu  11G nov  8 18:35 w2016.qcow2
-rw-r--r-- 1 root         root         1,3G nov  5 17:39 w2016_server.qcow2
-rw-r--r-- 1 libvirt-qemu libvirt-qemu 9,1G nov  8 18:41 win10.qcow2
```

Podemos observar que aunque el espacio de datos del volumen es solo de 3,32 MiB, el uso de espacio en disco se muestra como 21GiB, sin embargo si ejecutamos

```
du -h /var/lib/libvirt/images/guest1-deb9.qcow2
```

La salida es

```
3,4M    /var/lib/libvirt/images/guest1-deb9.qcow2
```

Esto quiere decir que el volumen realmente ocupa el espacio en el sistema de archivos de los datos que contiene.

Si por algún motivo necesitamos que el archivo de volumen se muestre en el sistema de archivos con el espacio real ocupado por el mismo, es decir, de modo que el comando ls muestre el espacio real ocupado, podemos ejecutar el procedimiento de conversión de imágenes que veremos a continuación

Convertir imágenes

Ejecutemos a través de los comandos siguientes la conversión de la imagen anterior

```
cd /var/lib/libvirt/images
qemu-img convert -O qcow2 guest1-deb9.qcow2 guest1-deb9.qcow2.new
rm  guest1-deb9.qcow2
mv guest1-deb9.qcow2.new guest1-deb9.qcow2
```

El comando **qemu-img** se utiliza para gestionar las imágenes, o volúmenes de datos, de almacenamiento. La opción **convert** se utiliza para conversiones entre formatos.

Ahora ejecutamos de nuevo un **ls -lh**

```
ls -lh /var/lib/libvirt/images
```

La primera línea de esa salida sería

-rw-r--r-- 1 root root 193K nov 9 17:39 guest1-deb9.qcow2

Por último, veamos de nuevo la salida del primer comando

```
virsh vol-list default --details
```

La primera línea de esa salida sería

guest1-deb9.qcow2 /var/lib/libvirt/images/guest1-deb9.qcow2 file 20,00 GiB 196,00 KiB

El comando **convert** permite especificar varias opciones interesantes, como la posibilidad de comprimir la imagen resultante, con la **opción -c**, o cifrar sus datos, con la opción **-o encryption**.

Podemos ver más opciones del comando qemu-img con **qemu-img --help**

Conversión de imágenes entre sistemas de virtualización diferentes

Otra opción muy interesante es la posibilidad de crear imágenes en un formato nuevo compatible con otros sistemas de virtualización, como VirtualBox o VMWare

Por ejemplo vamos a convertir una imagen en formato **qcow2** a formato **vdi** para hacerlo compatible con **VirtualBox**

```
qemu-img convert -f qcow2 -O vdi /var/lib/libvirt/images/win10_img.qcow2 /var/lib/libvirt/images/win10_img.vdi
```

Crearía una nueva imagen en formato .vdi a partir de la imagen original en qcow2, de este modo puede conectarse el volumen resultante como disco duro virtual de una máquina virtual de VirtualBox

Modificar el tamaño de almacenamiento de un volumen

Podemos aumentar de modo sencillo el espacio disponible en un volumen, sin afectar al uso de espacio en disco real del mismo, para ello usamos también el comando qemu-img

```
qemu-img resize /var/lib/libvirt/images/guest1-deb9.qcow2 +1GB
```

Incrementaría el espacio disponible del volumen en 1GB

```
virsh vol-info guest1-deb9.qcow2 default
```

Mostraría

```
Name:           guest1-deb9.qcow2
Type:           file
Capacity:       <u>'''21,00 GiB'''</u>
Allocation:     200,00 KiB
```

Como podemos ver en la salida anterior, la capacidad del volumen ha pasado de 20GiB a 21GiB

Es posible decrementar, shrink, el tamaño de una imagen, indicándolo con un símbolo - delante del parámetro tamaño del comando resize, sin embargo, es necesario reducir el tamaño de las particiones creadas en la imagen mediante alguna herramienta del guest, para que esa opción funcione.

Chequeo de una imagen de disco

Podemos usar el comando check

```
qemu-img check guest1-deb9.qcow2
```

Mostraría la información resultante de efectuar un chequeo a la image indicada.

Si todo va bien:

```
No errors were found on the image.
Image end offset: 327680
```

Hacer commit de cambios en una imagen

Cuando estamos utilizando imágenes enlazadas, en las que unas toman otras como base, podemos confirmar los cambios sobre la imagen base mediante el comando commit.

Supongamos que tenemos las imágenes

```
w2016.qcow2          /var/lib/libvirt/images/w2016.qcow2
w2016_server.qcow2   /var/lib/libvirt/images/w2016_server.qcow2
```

En este supuesto la imagen w2016_server.qcow2 fue generada tomando como archivo de almacenamiento base w2016.qcow2. Esta imagen almacenará las diferencias respecto al archivo de almacenamiento base. Si queremos incluir esas diferencias en ese archivo base podremos utilizar el comando commit.

Ejecutamos

```
qemu-img info w2016_server.qcow2
```

Mostraría

```
image: w2016_server.qcow2
file format: qcow2
virtual size: 20G (21474836480 bytes)
disk size: 2.2G
cluster_size: 65536
backing file: /var/lib/libvirt/images/w2016.qcow2
Snapshot list:
ID        TAG                 VM SIZE                DATE       VM CLOCK
2         snap1                  823M 2017-11-02 23:51:17   00:28:01.565
Format specific information:
    compat: 1.1
    lazy refcounts: false
    refcount bits: 16
    corrupt: false
```

Podemos observar como esa imagen tiene como backing file /var/lib/libvirt/images/w2016.qcow2, que corresponde con la otra imagen del primer listado. En este caso podríamos hacer un commit de esa imagen para guardar los cambios en el archivo de imagen principal, de modo que la base se verá actualizada con los cambios. Ejecutamos:

```
qemu-img commit w2016_server.qcow2
```

Comparar imágenes

Podemos utilizar para esto el comando compare

```
qemu-img compare w2016.qcow2 win10.qcow2
```

Mostraría en este caso un mensaje indicando que las imágenes difieren a partir del bloque indicado en la salida. Por defecto pueden comparse dos imágenes de distinto tamaño, pueden identificarse como iguales si el espacio adicional de una respecto a la otra solo contiene ceros.

Ver mapa de datos de una imagen

Con el comando map podemos ver los metadatos del archivo de imagen, más concretamente mostrará las cadenas de bytes y sus direcciones dentro de la imagen, además del archivo en el que están almacenadas

```
qemu-img map win10.qcow2 --output=json
```

Si no se especifica la opción --output mostrará un formato de salida ?humano?

Realizar un amend (enmendar) de las opciones de una imagen

Con el comando amend podemos enmendar opciones específicas del formato de la imagen. La sintaxis del comando sería

qemu-img ammend [-p] [-f formato] [-t cache] -o options filename

-p mostraría el progreso del comando

Cambiar el archivo de base de una imagen

Con el comando **rebase** podremos cambiar el archivo de base de una imagen.

Podemos ver las opciones de uso con

```
qemu-img --help
```

Listado, creación, borrado y aplicación de snapshots

Las spanshots, instantáneas, son una herramienta muy útil a la hora de trabajar con imágenes. Mediante su uso podremos recuperar el estado previo de la imagen almacenado en la snahpshot.

Para gestionar este aspecto disponemos del comando snapshot

La sintaxis del comando es simple

qemu-img snapshot [-l | -a snapshot | -c snapshot | -d snapshot] filename

Las opciones son evidentes* **-l** lista las snapshot de una imagen

- **-a** aplica la snapshot indicada a la imagen
- **-c** crea una snapshot con el nombre indicado
- **-d** elimina la snapshot de la imagen de nombre indicado

Vamos a crear una snapshot para una de las imágenes

```
qemu-img snapshot -c snap1 win10.qcow2
```

Crearía la snapshot de nombre snap1 para la imagen win10.qcow2

Vamos a comprobar que efectivamente se ha creado

```
qemu-img snapshow -l win10.qcow2
```

Mostraría

```
Snapshot list:
ID        TAG                 VM SIZE                DATE       VM CLOCK
1         snap1                     0 2017-11-09 19:18:59   00:00:00.000
```

Por último la borramos

```
qemu-img snapshot -d snap1 win10.qcow2
```

Volver

JavierFP 12:55 10 nov 2017 (CET)