Almacenamiento en KVM

Bibliografía:

Red Hat Enterprise Linux 7

Virtualization Deployment and Administration Guide

Contenidos

1. Objetivos.

2. Ciclo de vida de un contenedor de volúmenes (*storage pool*).

3. Manejo de volúmenes.

1. Objetivos

- Capacidad para realizar todas las tareas de gestión del almacenamiento en un sistema anfitrión.
  - Capacidad administración de los recursos a utilizar por los sistemas invitados (gestión de la infraestructura de almacenamiento).
  - Capacidad integrar estos recursos en los sistemas invitados (manejo de las configuraciones de los sistemas invitados).

2. El ciclo de vida de un contenedor de volúmenes (*storage pool*)

- Contenedor de almacenamiento (*storage pool*). Abstracción de un espacio de almacenamiento en el que se crearán los volúmenes virtuales de almacenamiento utilizados por las MV.
- Se puede crear sobre:
  - NFS
  - iSCSI
  - RBD
  - *Sheepdog cluster*
  - SCSI
  - Volúmenes LVM
  - Discos
  - Particiones
  - Directorios

2. El ciclo de vida de un contenedor de volúmenes (*storage pool*)

- Su uso no es obligado, pero se justifica por razones de seguridad.
  - Aislamiento del espacio utilizado por las MVs.
  - Uso de mecanismos específicos de seguridad.
- Ciclo de vida:
  1. Definición del contenedor de almacenamiento.
  2. Creación del contenedor de almacenamiento.
  3. Puesta en marcha/parada del contenedor de almacenamiento.
  4. Eliminación de un contenedor de almacenamiento

2. El ciclo de vida de un contenedor de volúmenes: Definición (I)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML:
  - Metadatos generales.
  - Elementos del origen.
  - Elementos del destino.
  - Extensiones.

2. El ciclo de vida de un contenedor de volúmenes: Definición (II)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML:

Metadatos generales	**Elementos del origen**	**Elementos del destino**
name	device	path
uuid	dir	permissions
allocation	adapter	timestamp
capacity	host	encryption
available	auth	
	name	
	vendor	
	product	

2. El ciclo de vida de un contenedor de volúmenes: Definición (III)

```
#  virsh pool-define-as name type \
    [--source-host hostname] \
    [--source-path path] \
    [--source-dev path] \
    [*--source-name name*] \
    [*--target path*] \
    [*--source-format format*] \
    [--source-initiator initiator-iqn] \
    [*--auth-type authtype* *--auth-username username* \
    [*--secret-usage usage* | *--secret-uuid uuid*]] \
    [*--source-protocol-ver ver*] \
    [[*--adapter-name name*] | [*--adapter-wwnn* *--adapter-wwpn*] \
    [*--adapter-parent parent*]] \
    [*--print-xml*]
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un directorio del sistema (I)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: Directorio (dir)

```
<pool type="dir">
   <name>virtimages</name>
   <target>
         <path>/var/lib/libvirt/images</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un directorio del sistema (II)

```
# virsh pool-define-as virtimages dir \
 –-target /var/lib/libvirt/images
```

2. El ciclo de vida de un contenedor de volúmenes: Un contenedor asociado a un sistema de archivos (I)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: Sistema de ficheros (fs): ext2, etx3, ext4, ufs, iso9660, udf, gfs, gfs2, vfat, hfs+, xfs, ocfs2

```
<pool type="fs">
   <name>virtimages</name>
   <source>
         <device path="/dev/VolGroup00/VirtImages"/>
   </source>
   <target>
         <path>/var/lib/libvirt/images</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Un contenedor asociado a un sistema de archivos (II)

```
# virsh pool-define-as virtimages fs  \
    --source-dev  /dev/VolGroup00/VirtImages \
    --target /var/lib/libvirt/images
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un grupo de volúmenes lógicos

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: Logical Volume (logical): lvm2

```
<pool type="logical">
    <name>HostVG</name>
    <source>
            <device path="/dev/sda1"/>
            <device path="/dev/sdb1"/>
            <device path="/dev/sdc1"/>
    </source>
    <target>
            <path>/dev/HostVG/</path>
    </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un sistema de archivos de red (I)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: Sistema de ficheros en red (NFS)

```
<pool type="netfs">
   <name>virtimages</name>
   <source>
         <host name="nfs.example.com"/>
         <dir path="/var/lib/libvirt/images"/>
   </source>
   <target>
         <path>/var/lib/libvirt/images</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un sistema de archivos de red (II)

```
# virsh pool-define-as virtimages netfs \
    --source-host nfs.example.com \
    --source-path /var/lib/libvirt/images \
    --target /var/lib/libvirt/images
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un disco (I)

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: Volumen de disco (disk): dos, dvh, gpt, bsd, pc98, sun, etc.

```
<pool type="disk">
   <name>sdb</name>
   <source>
         <device path="/dev/sdb"/>
   </source>
   <target>
         <path>/dev</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a un disco (II)

```
# virsh pool-define-as sdb disk \
    --source-dev /dev/sdb \
    --target /dev
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado una unidad iSCSI

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: iSCSI (iscsi)

```
<pool type="iscsi">
   <name>virtimages</name>
   <source>
         <host name="iscsi.example.com"/>
         <device path="demo-target"/>
   </source>
   <target>
         <path>/dev/disk/by-path</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Contenedor asociado a una unidad SCSI

```
# virsh pool-define archivo_definición_contenedor.xml
```

- Formato XML: SCSI (scsi)

```
<pool type="scsi">
   <name>virtimages</name>
   <source>
         <adapter name="host0"/>
   </source>
   <target>
         <path>/dev/disk/by-path</path>
   </target>
</pool>
```

2. El ciclo de vida de un contenedor de volúmenes: Creación

Formato de la orden:

```
# virsh pool-build Nombre_contenedor
```

Ejemplo:

```
# virsh pool-build virtimages
```

2. El ciclo de vida de un contenedor de volúmenes: Puesta en marcha

Formato de la orden:

```
# virsh pool-start Nombre_contenedor
# virsh pool-autostart Nombre_contenedor
```

Ejemplos:

```
# virsh pool-start virtimages ; virsh pool-autostart virt_images
```

2. El ciclo de vida de un contenedor de volúmenes: Eliminación

Se realiza en dos o tres pasos, dependiendo del tipo de contenedor.

- Paso1: evitar que el contenedor y los volúmenes que contengan sean utilizados por un sistema invitado.
- Paso 2 (opcional): eliminar el directorio asociado al contenedor.
- Paso 3: eliminar la definición del contenedor.

Formato de las ordenes:

```
# virsh pool-destroy Nombre_contenedor
# virsh pool-delete Nombre_contedor
# virsh pool-undefine Nombre_contedor
```

Dos ejemplos:

```
# virsh pool-destroy virtimages
# virsh pool-delete virtimages
# virsh pool-undefine virtimages
```

2. El ciclo de vida de un contenedor de volúmenes: Monitorizar el estado

Obtener los contenedores definidos

```
# virsh pool-list  --details | -all
```

Obtener información de un contenedor específico

```
# virsh pool-info  Nombre_Conteendor
```

3. Manejo de volúmenes:
Creación de volúmenes en contenedores

Formato de la orden:

```
# virsh vol-create-as Nombre_Cont Nombre_Volumen TamañoG
```

Ejemplos:

```
# virsh vol-create-as virtimages volumen1 10G
# virsh vol-create-as virtiamges volumen2 8G
```

3. Manejo de volúmenes:
Creación de volúmenes manualmente en contenedores

```
# dd if=/dev/zero of=/Mi_Contenedor/profeslab33.img \
    bs=1M count=4096
```

Alternativamente

```
 # dd if=/dev/zero of=/Mi_Contenedor/profeslab33.img \
    bs=1M count=0 seek=4096
```

OJO: Habría que añadir los atributos SElinux correspondientes

3. Manejo de volúmenes:
Obtener información de volúmenes ubicados en un contenedor

Formato de la orden:

```
# virsh vol-list Nombre_Contenedor
```

Ejemplo:

```
# virsh vol-list virtimages
```

3. Manejo de volúmenes:
Clonar volúmenes ubicados en contenedores basados en disco

Formato de la orden:

```
# virsh vol-clone --pool Nombr_Cont_Bas_Disc Volumen Volume_Clon
```

Ejemplo:

```
# virsh vol-clone –pool Contenedor_Disc volumen2 clonvol2
```

3. Manejo de volúmenes:
Añadir un volumen virtual de almacenamiento a una MV (I)

Paso 1. Crear el volumen.

Paso 2. Generar el archivo XML que describe el disco virtual

Disco_Virtual.xml

```
<disk type='file' device='disk'>
   <driver name='qemu' type='raw' cache='none'/>
   <source file='/Mi_Contenedor/profeslab33.img'/>
   <target dev='vdb'/>
</disk>
```

CDROM_Virtual.xml

```
<disk type='file' device='cdrom'>
     <driver name='qemu' type='raw' cache='none'/>
     <source file='/ImagnesDistroISO/centos/CentOS-6.4-x86_64-bin-DVD1.iso'/>
     <target dev='vdb'/>
</disk>
```

3. Manejo de volúmenes:
Añadir un volumen virtual de almacenamiento a una MV (II)

Paso 3. Conectar dispositivos a la MV

```
# virsh attach-device  --config  profeslab33  ~/Disco_Virtual.xml
# virsh attach-device  --config  profeslab33  ~/CDROM_Virtual.xml
```

Una vez conectado el nuevo volumen, para utilizarlo:

Arrancar la MV

```
# virsh start profeslab33
```

Desde la máquina virtual (pofeslab33) particionar disco duro

```
# fdisk /dev/vdb
```

Desde la máquina virtual (pofeslab33) crear un sistema de archivos

```
# mkfs –ext3 /dev/vdb1
```

Desde la máquina virtual (pofeslab33) Montar el sistema de archivos

```
# mount   /dev/vdb1  /VDB1
```

3. Manejo de volúmenes:
Añadir un volumen virtual de almacenamiento a una MV (III)

Alternativa a los pasos 2 y 3 anteriores:

```
# virsh attach-device-as profeslab33 \
    /Mi_Contenedor/profeslab33.img vdb –-targetbus virtio
    --config
```

3. dispositivos físicos:
Añadir dispositivo físico de bloque (disco duro, CDROM, DVD) a una MV (I)

Los pasos serían análogos al caso anterior, teniendo en cuenta que:

- El paso 1 del caso anterior no sería necesario.
- El archivo XML que describe al volumen físico sería distinto. Por ejemplo, si se quisiera añadir el disco del sistema anfitrión `/dev/sdc`:

Disco_Fisico.xml

```
<disk type=‘block' device='disk'>
  <driver name='qemu' type='raw' cache='none'/>
  <source  dev=‘/dev/sdc'/>
  <target dev='vdc'/>
</disk>
```

3. dispositivos físicos:
Añadir dispositivo físico de bloque (disco duro, CDROM, DVD) a una MV (II)

Alternativamente:

```
# virsh attach-disk  profeslab33 /dev/scd  vdc  --config
```