ADMINISTRANDO VIRTUALIZACIÓN KVM CON VIRSH

Virsh, abreviatura de Virtual Shell, es una interfaz CLI para administrar máquinas virtuales invitadas. Permite crear, enumerar, editar, iniciar, reiniciar, detener, suspender, reanudar, apagar y eliminar máquinas virtuales. Actualmente es compatible con KVM, LXC, Xen, QEMU, OpenVZ, VirtualBox y VMware ESX.

Claramente que para usarlo haz tenido que instalar KVM y virsh en tu distro Gnu/Linux de cabecera, ademas de ser configurado correctamente. Voy a obviar este paso, ya que escribí una entrada al respecto y que podría ser de ayuda, visita lo en el siguiente link instalando qemu como virtualizador.

Sí es tu primera vez usando virsh, o KVM, darle una ojeada al help u manual del comando es una buena practica para conocerlo, para el caso de virsh, basta lanzar el comando:

```
$ virsh help
```

Que devolverá una lista extensa de comandos virsh disponibles para administrar máquinas virtuales KVM desde la línea de comandos. Simplemente lea la descripción de un comando que desea ejecutar y utilícelo, no es vital memorizar todo de una sola, la practica ayuda mucho. Esta nueva entrada es más avanzada que la anterior, así que recomiendo darle una mirada.

Los comandos se agrupan en las siguientes secciones:

```
    Domain Management,
    Domain Monitoring,
    Host y Hypervisor,
    Checkpoint,
    Interface,
    Network Filter,
    Networking,
    Node Device (Dispositivo de nodo),
    Secret,
    Snapshot (Instantánea),
    Backup (Respaldo,
    Storage Pool,
    Storage Volume,
    Virsh.
```

Cada sección contiene los comandos relacionados para realizar un conjunto particular de tareas. Puede ver la sección de ayuda de un grupo, por ejemplo, **Networking**, como se muestra a continuación:

```
$ virsh help Networking
...
...
net-autostart autoinicia una red
net-create crea una red a partir de un archivo XML
net-define define una red virtual persistente inactiva o modifica
una persistente existente desde un archivo XML
...
...
```

También puede mostrar la sección de ayuda de un comando específico. Como:

```
$ virsh help net-name

NAME
net-name: convierte un UUID de red en un nombre de red
...
```

Listar, Renombrar e Iniciar máquinas virtuales

```
$ sudo virsh net-list --all
```

Con el comando que inicia esta sección, puedes averiguar todas las VM's disponibles y con un simple **start** podrás iniciar una u otra maquina, por ejemplo supongamos que tenemos una con el nombre **AlmaLinux** y para lanzarla ejecuta:

```
$ virsh start AlmaLinux
Domain AlmaLinux started
```

Para verificar si la VM se está ejecutando, use el comando **list**:

```
$ virsh list

Id       Name         State
------------------------------
1      AlmaLinux     running
```

El comando virsh domrename se utiliza para cambiar el nombre de un dominio. Este comando cambia el nombre de dominio actual por el nuevo nombre sin realizar ningún cambio en los archivos de configuración.

```
$ virsh domrename nombre_actual nuevo_nombre
```

Tenga en cuenta que el dominio debe estar inactivo y sin instantáneas o puntos de control.

Guardar y Restaurar máquinas virtuales

Para guardar el estado actual de una máquina virtual en ejecución, puedes ejecutar save, así:

```
$ virsh save AlmaLinux almalinux-save
Domain AlmaLinux saved to almalinux-save
```

Este comando detiene al guest y guarda los datos en un archivo llamado **almalinux-save**. Para restaurar el estado guardado previamente de una máquina virtual, simplemente especifique el nombre del archivo como se muestra a continuación:

```
$ virsh restore almalinux-save
Domain restored from almalinux-save
```

Reiniciar y Suspender/Pausar máquinas virtuales

Para reiniciar una máquina virtual en ejecución, ejecute:

```
$ virsh reboot AlmaLinux
...
```

Sí quieres suspender una máquina virtual, basta con:

```
$ virsh suspend AlmaLinux
...
```

Este estado es considerado como pausado, también esta el comando resume, que nos ayuda a reactivar, a reanudar la VM, saliendo de la pausa:

```
$ virsh resume AlmaLinux
...
```

Detener y Apagar máquinas virtuales activas

Para detener a la fuerza una máquina virtual activa y dejarla en estado inactivo, ejecuta un destroy:

```
$ virsh destroy AlmaLinux
...
```

También puedes detener de mejor manera la máquina virtual en lugar de forzarla como se muestra a continuación, gracias a la opción **–graceful**:

```
$ virsh destroy AlmaLinux --graceful
....
```

Mientras que para apagar una máquina virtual en ejecución, usarías **shutdown**:

```
$ virsh shutdown AlmaLinux
...
```

Crear y Recuperar maquinas virtuales con volcado XML

Puede crear una nueva máquina virtual invitada utilizando el XML existente de invitados creados anteriormente. Primero, cree un volcado XML como se muestra arriba y luego cree una nueva VM usando el archivo XML como se muestra a continuación:

```
$ virsh create AlmaLinux.xml
...
```

Este comando creará una nueva máquina virtual y la iniciará de inmediato. Puedes verificarlo usando el comando `virsh list`, arriba mencionado. Para mostrar el archivo de configuración XML de una máquina virtual en la salida estándar, ejecute:

```
$ virsh dumpxml Almalinux
...
```

Mostrando los detalles completos de la configuración (software y hardware) de la máquina virtual. Tambien puedes exportar el volcado XML a un archivo en lugar de simplemente mostrarlo en la salida estándar

```
$ virsh dumpxml AlmaLinux > AlmaLinux.xml
```

El archivo xml, se guardará en el directorio de trabajo actual. Si deseas realizar cambios en las guest, simplemente puedes editar su archivo de configuración y realizar los cambios que desee; tambien se pueden editar mientras corren o mientras están fuera de línea.

```
$ virsh edit AlmaLinux
```

Este comando abrirá el archivo en su editor predeterminado que configuró con la variable $EDITOR.

Habilitar el acceso a la consola para máquinas virtuales

Después de crear máquinas invitadas KVM, puedes acceder a ellas a través de SSH, cliente VNC, Virt-viewer, Virt-manager, etc. Sin embargo, no puedes acceder a ellas usando el comando virsh console (se utiliza para conectar la consola serie virtual para el invitado). Para acceder a los invitados de KVM mediante esta, debes habilitar el acceso a la consola en serie en la máquina invitada.

Debes agregar una consola serie en su máquina invitada para acceder a su consola virsh desde el sistema host. Para hacerlo, inicia sesión en su máquina invitada a través de SSH o Virt-manager, ejecutando los siguientes comandos para habilitar e iniciar una consola en serie:

```
# systemctl enable serial-getty@ttyS0.service
# systemctl start serial-getty@ttyS0.service
```

Tener en cuenta que los comandos anteriores deben ejecutarse en el sistema invitado KVM (máquina virtual), no en el host KVM. Puedes verificar mirando en el archivo XML de configuración de la máquina virtual. Mostrando algo similar a esto:

```
<serial type='pty'>
    <target type='isa-serial' port='0'>
         <model name='isa-serial'/>
    </target>
</serial>
<console type='pty'>
    <target type='serial' port='0'/>
</console>
```

Ahora inicie la consola virsh del sistema invitado desde el host usando el comando:

```
$ virsh console AlmaLinux
Connected to domain AlmaLinux
Escape character is ^]
```

Presiona ENTER nuevamente y escriba su nombre de usuario y contraseña para conectarse a la máquina invitada Ahora está dentro de la consola de la máquina invitada. Empiece a usarlo. Puede volver a la consola del host en cualquier momento presionando las teclas CTRL +].

Mostrar detalles de máquinas virtuales y del host

Para mostrar la información de una máquina invitada, use el nombre de dominio (domname), la identificación de dominio (domid) o la uuid de dominio (domuuid), en conjunto a dominfo como a continuación:

```
$ virsh dominfo AlmaLinux
$ virsh dominfo 2
$ virsh dominfo de4100c4-632e-4c09-8dcf-bbde29170268
```

Para obtener la información de su sistema host, ejecute:

```
$ virsh nodeinfo
```

Para mostrar la información de la CPU virtual, ejecute:

```
$ virsh vcpuinfo AlmaLinux
```

Eliminar máquinas virtuales

Si ya no desea una máquina virtual, simplemente puedes eliminara de la siguiente manera:

```
$ virsh destroy AlmaLinux

$ virsh undefine AlmaLinux
```

El primer comando detendrá a la fuerza la máquina virtual si ya se está ejecutando. Y el segundo comando lo anulará y eliminará por completo. También se puede utilizar las siguientes opciones para eliminar los volúmenes de almacenamiento y las instantáneas.

```
--managed-save eliminar el archivo de estado administrado del
dominio
--storage elimina los volúmenes de almacenamiento asociados (lista
separada por comas de destinos o rutas de origen) (ver domblklist)
--remove-all-storage elimina todos los volúmenes de almacenamiento
asociados (usarlo con precaución, teniendo claro los volúmenes)
--delete-storage-volume-snapshots eliminar instantáneas asociadas
con los volúmenes
--wipe-storage borra los datos de los volúmenes eliminados
--snapshots-metadata elimina todos los metadatos de instantáneas del
dominio (vm inactiva)
```

Administrar redes virtuales

Vista la gestión de VM's, damos un salto a las redes virtuales siempre con virsh. Para listar las redes virtuales disponibles, ejecute:

```
$ virsh net-list

 Name      State    Autostart   Persistent
```

```
 ----------------------------------------------
  default   active   yes         yes
```

Como puede ver, solo tengo una red virtual que es la predeterminada. Para ver los detalles de una red virtual, ejecute:

```
$ virsh net-dumpxml default
```

default, se debe reemplazar con el nombre de su red, dependiendo de tu caso. Si quieres iniciar una red inactiva, ejecuta:

```
$ virsh net-start nombre_de_red_inactiva
```

Y para que se inicie automáticamente una red:

```
$ virsh net-autostart nombre_de_red
```

También como con las VM's, puedes crear volcados XML de redes virtuales. Para crear el archivo de configuración XML de una red virtual existente, ejecuta:

```
$ virsh net-dumpxml default > default.xml
```

Puedes ver el archivo XML usando el comando cat:

```
$ cat default.xml
<network connections='1'>
  <name>default</name>
  <uuid>ce25d978-e455-47a6-b545-51d01bcb9e6f</uuid>
  <forward mode='nat'>
    <nat>
      <port start='1024' end='65535'/>
    </nat>
  </forward>
  <bridge name='virbr0' stp='on' delay='0'/>
  <mac address='52:54:00:ee:35:49'/>
```

```
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254'/>
    </dhcp>
  </ip>
</network>
```

Para crear una nueva red virtual utilizando un archivo XML existente e iniciarlo inmediatamente, ejecuta:

```
$ virsh net-create nombre_archivo_xml
```

Si deseas crear una red a partir de un archivo XML pero no desea iniciarla automáticamente, ejecuta:

```
$ virsh net-define nombre_archivo_xml
```

Para desactivar una red activa, ejecute:

```
$ virsh net-destroy nombre_red
```

Para eliminar una red virtual, desactíva primero como se muestra arriba y luego ejecuta:

```
$ virsh net-undefine nombre_red_inactiva
```

Como vez muchos de los comandos de networking, se asemejan a los de manejo de las Vm's, la diferencia el prefijo net. 😉

Conclusión

Aprender a usar la herramienta CLI de Virsh a fondo es suficiente para configurar un entorno virtual completo en Linux. No necesita ninguna aplicación GUI. Pero claro, si no eres de los que aman la CLI puedes probar las herramientas de administración de KVM gráficas como Virt-manager y Cockpit. Conociendo virsh estás a medio

camino para administrar un entorno de virtualización de nivel empresarial. La configuración de KVM y la gestión de máquinas virtuales KVM mediante el comando virsh son muy importantes para todos los Linux SysAdmin.

Hasta otro post lector, buenas vibras. Happy Hacking!

Fuente

virsh

man virsh

Posted in GNU/Linux, SysAdmin and tagged CLI, KVM, Qemu, shell, SysAdmin, virsh, Virtualización on 29 julio, 2021. Deja un comentario

← Agregando Fuentes Nuevas en Fedora

Keybindings útiles en Gnu/Linux para la terminal →