Tecnologías Especificas de la Ingeniería Informática

Curso 2021/2022

Prácticas de Virtualización

Grado Ingeniería Informática

Universidad de Murcia

Primeros pasos

Descargar imágenes (.iso)

```
wget http://cdimage.ubuntu.com/releases/18.04.4/release/ubuntu-18.04.4-server-amd64.iso
```

VirtualBox

Listar VMs

```
VBoxManage list vms
```

Mostrar Info VMs

```
VBoxManage showvminfo Linux-Ubuntu
```

Crear VM

```
VBoxManage createvm --name Linux-Ubuntu --ostype Ubuntu_64 --register
```

Añadir RAM a la VM

```
VBoxManage modifyvm Linux-Ubuntu --memory 2048
```

Crear un disco virtual

```
VBoxManage createhd --filename /home/alumno/disco.vdi --size 1024
```

Añadir el disco a la VM

```
VBoxManage storagectl Linux-Ubuntu --name "SATA Controller" --add sata --controller IntelAhci

>VBoxManage storageattach Linux-Ubuntu --storagectl "SATA Controller" --type HDD --port 1 --device 0 --medium /home/alumno/disco.vdi
```

Añadir el CDROM

```
>VBoxManage storagectl Linux-Ubuntu --name "IDE Controller" --add ide --controller PIIX4

> VBoxManage storageattach Linux-Ubuntu --storagectl "IDE Controller" --type DVDDrive --port 1 --device 0  --medium /home/alumno/ubuntu-18.04.4-server-amd64.iso
```

Listar discos existentes

```
VBoxManage list hdds
```

Mostrar información de un disco

```
VBoxManage showmediuminfo /home/alumno/disco.vdi
```

Cambiar tamaño disco

```
VBoxManage modifyhd 08fc0309-1895-46e9-ad01-5d83acee833b --resize 15000
```

Lanzar una Máquina virtual

```
VBoxHeadless --startvm Linux-Ubuntu
```

```
VBoxManage startvm Linux-Ubuntu --type headless
```

(headless para no abrir la ventana gráfica)

Pausar VM

```
VBoxManage controlvm Linux-Ubuntu pause
```

Resume VM

```
VBoxManage controlvm Linux-Ubuntu resume
```

Apagar VM

```
VBoxManage controlvm Linux-Ubuntu poweroff
```

Clonar VM

```
VBoxManage clonevm Linux-Ubuntu --name Cloned-VM --register
```

Eliminar VM

```
VBoxManage unregistervm Cloned-VM --delete
```

Exportar VM

```
VBoxManage export Linux-Ubuntu --output maqexportada.ova
```

Convertir la imagen a formato Raw (compatible con KVM)

```
VBoxManage clonehd --format RAW WindowsXP.vdi WindowsXP.raw
```

GESTIÓN DE MÁQUINAS VIRTUALES CON LIBVIRT - VIRSH

virsh es una interfaz de comandos para gestionar máquinas virtuales. En la interfaz virsh, las máquinas virtuales se identifican por sus nombres de dominio, por lo que virsh generalmente se usa para enumerar los dominios actuales, para crear, pausar y cerrar dominios. Virsh es compatible con Xen, QEmu, KVM, LXC, OpenVZ, VirtualBox y VMware ESX.

Ejemplo: 1 Obtener la versión instalada

```
alumno@lab23:~$ virsh version
Compiled against library:     libvirt 4.0.0
Using library:                libvirt 4.0.0
Using API:                    QEMU 4.0.0
Running hypervisor:           QEMU 2.11.1
```

Ejemplo: 2 obtener la memoria del Hypervisor(Host)

```
alumno@lab23:~$ virsh nodememstats
total  :             16327600 KiB
free   :             10372152 KiB
buffers:               558780 KiB
cached :              2245096 KiB
```

Ejemplo: 3 obtener la información de CPU del Hypervisor

```
alumno@lab23:~$ virsh nodecpustats
user:                  1162410000000
system:                 361170000000
idle:                  7297160000000
iowait:                  16650000000
```

Ejemplo: 4 Crear una nueva VM usando virt-install

- Crear disco de 10Gb

```
sudo qemu-img create -f qcow2 /var/lib/libvirt/images/ubuntu-linux.img
10G
```

-Comando **virt-install** para crear la VM (añadiéndole el disco creado antes)

```
virt-install --virt-type kvm --name ubuntu-linux --ram 2048 --vcpus=2  --
disk path=/var/lib/libvirt/images/ubuntu-linux.img --network
bridge=virbr0 --graphics vnc --os-variant=ubuntu18.04 --cdrom
/home/alumno/ubuntu-18.04.4-server-amd64.iso
```

Ejemplo: 5 Obtener las máquinas virtuales y su estado:

```
[root@localhost ~]# virsh list --all
Id Name              State
----------------------------------
3 test               running
```

Ejemplo: 6 Obtener las redes disponibles del KVM hypervisor

```
[root@localhost ~]# virsh net-list
Name                 State      Autostart
-----------------------------------------
default              active     yes
```

Ejemplo: 7 Obtener información de hardware de una VM

Synatx : virsh dominfo BaseMachine

```
[root@localhost ~]# virsh dominfo test
Id:             3
Name:           test
UUID:           9ae96029-6c3d-8bd1-6e19-926183f89074
OS Type:        hvm
State:          running
CPU(s):         4
CPU time:       26862.0s
Max memory:     4194304 kB
Used memory:    4194304 kB
Persistent:     yes
Autostart:      disable
Managed save:   no
```

Ejemplo: 8 Apagar Virtual Machine

```
[root@localhost ~]# virsh shutdown machine_name
```

Ejemplo: 9 Reiniciar the Virtual Machine

```
[root@localhost ~]# virsh reboot machine_name
```

Ejemplo: 10 Forzar apapado o destruir la VM

```
[root@localhost ~]# virsh destroy machine_name
```

Ejemplo:11 Iniciar la VM

```
[root@localhost ~]# virsh start machine_name
```

Ejemplo:12 Conectar a una VM usando virt-viewer

syntax : virt-viewer -c qemu:///system machine_name

```
[root@localhost ~]# virt-viewer -c qemu:///system  test
```

Ejemplo 13: clonar

```
virt-clone --connect=qemu:///system -o nameVM -n newVMhost -f
/path/to/newhost.qcow2
```

Ejemplo 14: crear VM desde fichero de configuración

```
virsh create configuration_file.xml
virsh define configuration_file.xml
```

Diferencia: ***create*** crea máquinas temporales, que son eliminadas tras apagarlas. Si utilizamos ***define***, las máquinas (o dominios, según la nomenclatura virsh) persisten tras apagarlas o destruirlas, por lo que pueden volver a ser iniciadas (*o clonadas*).

```
<domain type='kvm'>
<name>Ubuntu-18</name>
<uuid>4b4c19e8-9d76-0c9dcbf8-12141823d393</uuid>
<memory>1048576</memory>
<currentMemory>1048576</currentMemory>
<vcpu>2</vcpu>

<os>
 <type arch='x86_64' machine='pc-0.12'>hvm</type>
  <boot dev='cdrom'/>
  <boot dev='hd'/>
  <bootmenu enable='no'/>
</os>

<features>
```

```
 <acpi/>
 <apic/>
 <pae/>
 </features>

 <clock offset='utc'/>
 <on_poweroff>preserve</on_poweroff>
 <on_reboot>restart</on_reboot>
 <on_crash>restart</on_crash>

 <devices>
 <emulator>/usr/bin/kvm</emulator>

 <disk type='file' device='disk'>
   <driver name='qemu' type='qcow2'/>
   <source file='/home/alumno/disk.qcow2'/>
   <target dev='hda' bus='ide'/>
   <address type='drive' controller='0' bus='0' unit='0'/>
 </disk>

<disk type='file' device='cdrom'>
 <driver name='qemu' type='raw'/>
 <target dev='hdc' bus='ide'/>
 <readonly/>
 <address type='drive' controller='0' bus='1' unit='0'/>
 </disk>

 <controller type='ide' index='0'>
   <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x1'/>
 </controller>

<interface type='network'>
 <mac address='52:54:00:4a:9a:02'/>
 <source network='default'/>
 <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
 </interface>

 <serial type='pty'>
 <target port='0'/>
 </serial>

<console type='pty'>
 <target type='serial' port='0'/>
 </console>
 <input type='mouse' bus='ps2'/>
```

```
<graphics type='vnc' port='-1' autoport='yes'/>
<sound model='ac97'>
<address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</sound>

<video>
<model type='cirrus' vram='9216' heads='1'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
</video>

<memballoon model='virtio'>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
</memballoon>
</devices>
</domain>
```

Ejemplo 15: data dump de un guest (dominio) [Crear el fichero *xml* asociado a una mv]:

```
$ virsh dumpxml [domain-id, domain-name o domain-uuid] >
<domain>.xml
```

virsh dumpxml Ubuntu-18 > ubuntu-exportada.xml

Ejemplo 16: Convertir la imagen raw en qcow2

```
qemu-img convert -f raw disk.raw -O qcow2 disk.qcow2
```

PISTA: `-f indica formato fichero origen, -O formato fichero salida.`

-convertir el vdi directamente a qcow2

```
$ qemu-img convert -p -f vdi -O qcow2 /home/alumno/disco.vdi
/home/alumno/disco.qcow2
```

-Para ver las IPs de la maquina

```
arp -n
```

https://libvirt.org/drvvbox.html