QEMU/KVM y libvirt

Instalación, gestión y almacenamiento

📧 José Domingo Muñoz

🏫 IES Gonzalo Nazareno · Dos Hermanas

📚 IV · Infraestructura Virtual

01

Introducción a QEMU/KVM + libvirt

Instalación, red y almacenamiento por defecto

Instalación de QEMU/KVM + libvirt

```
root@kvm:~# apt install qemu-system libvirt-clients libvirt-daemon-system

root@kvm:~# adduser usuario libvirt

usuario@kvm:~$ virsh -c qemu:///system list
```

Para no tener que indicar la conexión en cada comando, podemos definir una variable de entorno:

```
export LIBVIRT_DEFAULT_URI='qemu:///system'
```

ℹ️ Con la variable definida, todos los comandos `virsh` usarán `qemu:///system` sin especificarlo.

Red disponible por defecto

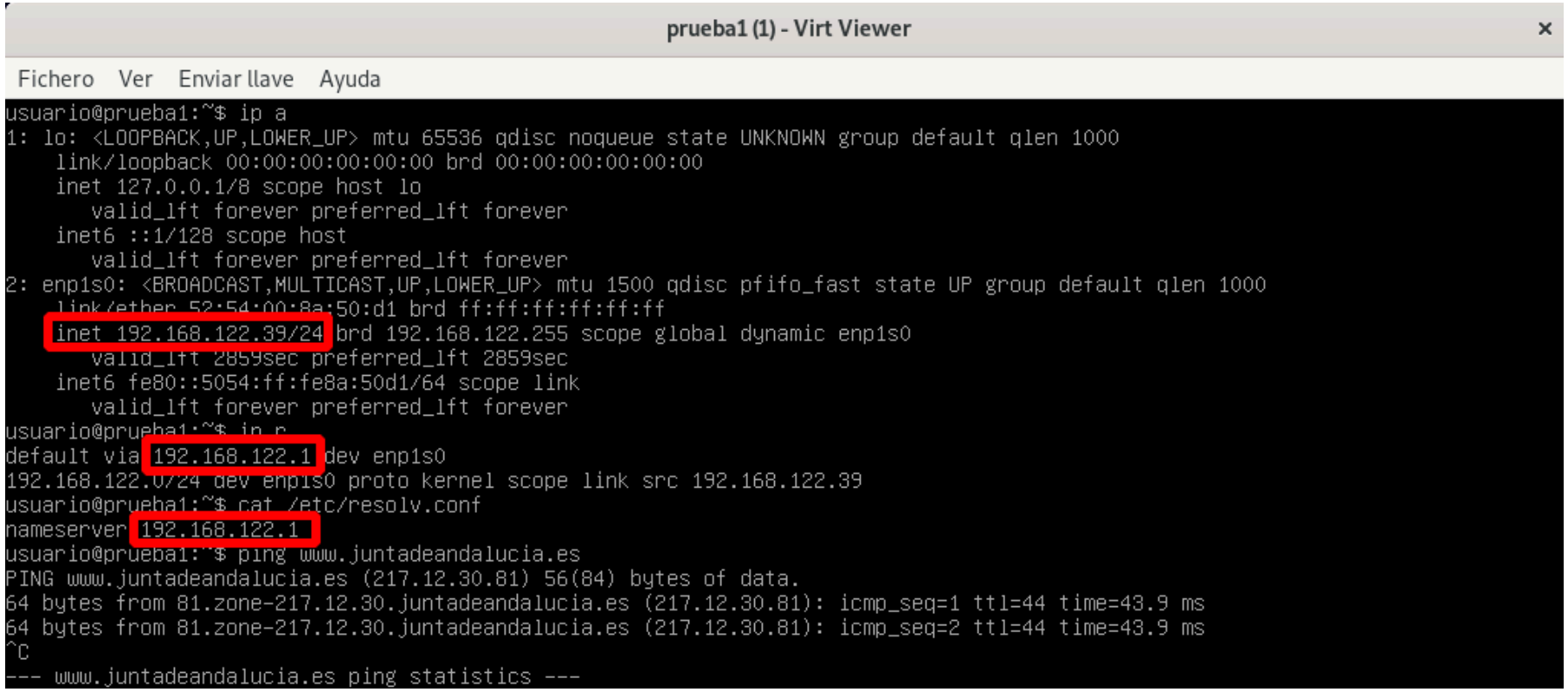
```
usuario@kvm:~$ virsh net-list --all
 Nombre    Estado     Inicio automático   Persistente
----------------------------------------------------
 default   inactivo   no                  si

usuario@kvm:~$ virsh net-start default
La red default se ha iniciado

usuario@kvm:~$ virsh net-autostart default
La red default ha sido marcada para iniciarse automáticamente
```

Características de la red `default`

La red `default` es de tipo **NAT**. Al crear una MV se conectará a ella por defecto:

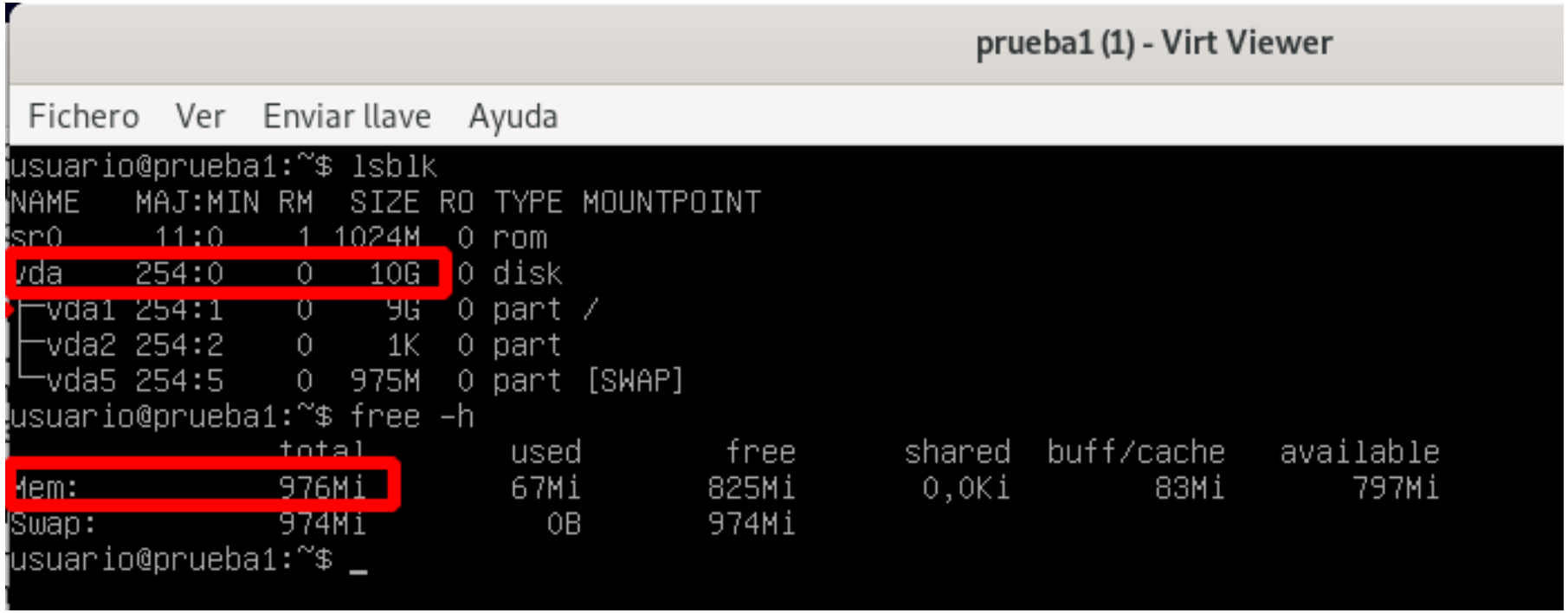
1. El host tiene un **servidor DHCP** (rango: `192.168.122.2 - 192.168.122.254`)
2. La **puerta de enlace** de la MV es `192.168.122.1` (el propio host)
3. El **servidor DNS** de la MV también es el host
4. La MV se conecta a un **Linux Bridge** llamado `virbr0`
5. El host se conecta al bridge `virbr0` con la IP `192.168.122.1`
6. El host realiza **SNAT** para dar conectividad al exterior

Red `default` — esquema



Almacenamiento disponible por defecto

- Los discos de las MV se guardan por defecto en **ficheros con formato** `qcow2`
- El directorio de almacenamiento es `/var/lib/libvirt/images`



Pools y volúmenes

POOL DE ALMACENAMIENTO

Recurso de almacenamiento gestionado por libvirt. Normalmente es un **directorio**.

```
virsh pool-list
 Nombre    Estado   Inicio automático
-----------------------------------------
 default   activo   si
 iso       activo   si
```

VOLUMEN

Medio de almacenamiento creado en un pool. Si el pool es de tipo `dir`, el volumen es un **fichero de imagen**.

```
virsh vol-list default
 Nombre          Ruta
-----------------------------------------------------
 prueba1.qcow2   /var/lib/libvirt/images/prueba1.qcow2
```

virt-install

```
apt install virtinst
```

Ejemplo — crear una MV con instalación desde ISO:

```
virt-install \
             --virt-type kvm \
             --name prueba1 \
             --cdrom ~/iso/debian-11.3.0-amd64-netinst.iso \
             --os-variant debian10 \
             --disk size=10 \
             --memory 1024 \
             --vcpus 1
```

Para acceder a la consola gráfica:

Gestión de MV con virsh

```
virsh --help
virsh list --help
```

Subcomandos más usados:

```
list --all                      dominfo <máquina>
shutdown <máquina>              domifaddr <máquina>
start <máquina>                 domblklist <máquina>
autostart <máquina>
reboot <máquina>
destroy <máquina>
suspend <máquina>
resume <máquina>
undefine --remove-all-storage <máquina>
```

Definición XML de una máquina

```
virsh dumpxml <máquina>
```

```
<domain type='kvm' id='6'>
  <name>prueba1</name>
  <uuid>a88eebdc-8a00-4b9d-bf48-cbed7bb448d3</uuid>
  ...
  <memory unit='KiB'>1048576</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
  <vcpu placement='static'>1</vcpu>
  ...
  <os>
    <type arch='x86_64' machine='pc-q35-5.2'>hvm</type>
    <boot dev='hd'/>
  </os>
```

Definición XML — disco e interfaz de red

```
<disk type='file' device='disk'>
  <driver name='qemu' type='qcow2'/>
  <source file='/var/lib/libvirt/images/prueba1.qcow2'/>
  <target dev='vda' bus='virtio'/>
  <address type='pci' domain='0x0000' bus='0x04' slot='0x00' function='0x0'/>
</disk>
...
<interface type='network'>
  <mac address='52:54:00:8a:50:d1'/>
  <source network='default'/>
  <model type='virtio'/>
  <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
</interface>
```

ℹ️ El disco y la tarjeta de red usan el bus **virtio** para mayor rendimiento (dispositivos paravirtualizados).

Modificación de una máquina virtual

EDICIÓN XML DIRECTA

```
virsh edit prueba1
```

Abre el XML en `$EDITOR`. Útil para cambios no soportados por comandos.

COMANDOS VIRSH

```
# Renombrar (MV parada)
virsh domrename prueba2 prueba1

# Cambiar vCPUs (MV parada)
virsh setvcpus prueba1 2 --config
```

⚠️ Algunos cambios requieren la MV **parada**, otros admiten **cambio en caliente** y otros necesitan **reinicio**.

Modificación de memoria

Con la MV **parada**, editando el XML:

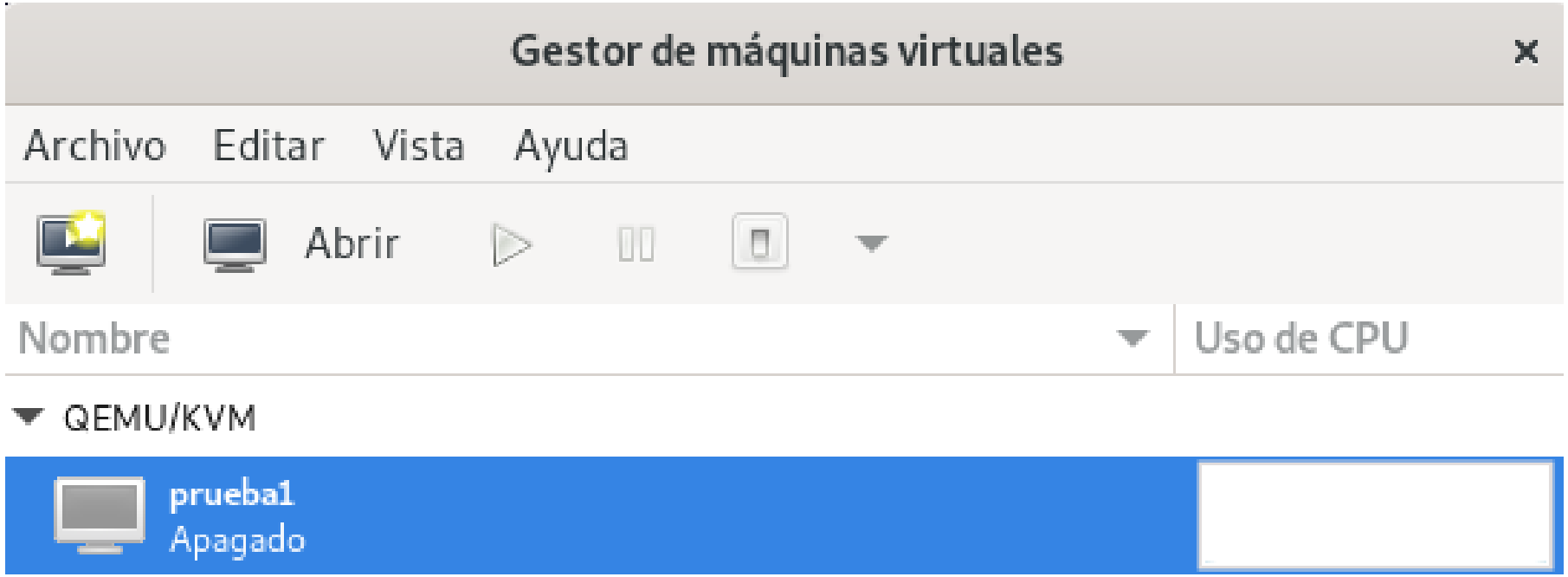
```
virsh edit prueba1
...
  <memory unit='KiB'>3145728</memory>
  <currentMemory unit='KiB'>1048576</currentMemory>
...
```

O **en caliente** con la MV arrancada:

```
virsh start prueba1

virsh setmem prueba1 2048M
```

virt-manager

Aplicación gráfica para gestionar libvirt:



Creación de MV Windows

- Configurar disco y tarjeta de red en modo **VirtIO**
- Windows **no tiene soporte nativo** para dispositivos VirtIO
- Añadimos un CDROM adicional con la **ISO de drivers VirtIO**

```
virt-install \
             --virt-type kvm \
             --name prueba4 \
             --cdrom ~/iso/Win10_21H2_Spanish_x64.iso \
             --os-variant win10 \
             --disk size=40,bus=virtio \
             --disk ~/iso/virtio-win-0.1.217.iso,device=cdrom \
             --network=default,model=virtio \
             --memory 2048 \
             --vcpus 2
```

02

Almacenamiento en QEMU/KVM + libvirt

Pools, volúmenes y gestión con virsh y qemu-img

Conceptos de almacenamiento

POOL DE ALMACENAMIENTO

Recurso de almacenamiento gestionado por libvirt.

Tipos principales:

- `dir` — directorio del sistema de archivos
- `logical` — grupo de volúmenes LVM
- `netfs` — directorio NAS (NFS...)
- `iSCSI` — disco desde servidor iSCSI

VOLUMEN

Medio de almacenamiento que representa el **disco de una MV**.

Según el tipo de pool, puede ser:

- Un **fichero de imagen** (`dir` , `netfs`)
- Un **volumen lógico LVM** (`logical`)
- Un **disco iSCSI** (`iscsi`)

Tipos de pool: `dir`

- El pool controla un **directorio del host**
- Los volúmenes son **ficheros de imagen de disco**:

FORMATO	CARACTERÍSTICAS
raw	Imagen binaria. Ocupa todo el espacio asignado. Más eficiente. Sin snapshots
qcow2	Copy-on-write. Aprovisionamiento ligero. Soporta snapshots. Algo menos eficiente
vdi, vmdk	Formatos de otros hipervisores (VirtualBox, VMware)

⚠️ El tipo `dir` **no ofrece almacenamiento compartido** entre hosts.

Tipos de pool: `logical`, `netfs`, `iSCSI`

`LOGICAL`

- Controla un **Grupo de Volúmenes LVM**
- Los volúmenes son **LVs**
- Sin almacenamiento compartido
- Sin snapshots ni aprovisionamiento ligero

`NETFS`

- Monta un **directorio NAS** (NFS...)
- Volúmenes: **ficheros de imagen**
- **Almacenamiento compartido** entre hosts

`ISCSI`

- Monta un **disco desde un servidor iSCSI**
- Los datos se guardan en ese disco remoto
- **Almacenamiento compartido** (con las consideraciones de acceso concurrente)

Gestión de volúmenes — dos enfoques

CON LIBVIRT (`VIRSH` / `VIRT-MANAGER`)

- Pool `dir` → crea una **imagen de disco**
- Pool `logical` → crea un **LV**

CON HERRAMIENTAS ESPECÍFICAS

- Pool `dir` → `qemu-img create …` y luego `pool-refresh`
- Pool `logical` → `lvcreate …` y luego `pool-refresh`

ℹ️ Tras crear el volumen con herramientas externas hay que ejecutar `virsh pool-refresh <pool>` para que libvirt lo detecte.

Gestión de pools de almacenamiento

```
virsh pool-list
virsh pool-info default
virsh pool-dumpxml default

# Crear, construir, arrancar y persistir un nuevo pool
virsh pool-define-as vm-images dir --target /srv/images
virsh pool-build vm-images
virsh pool-start vm-images
virsh pool-autostart vm-images

# Detener y eliminar
virsh pool-destroy vm-images
virsh pool-delete vm-images
virsh pool-undefine vm-images
```

Gestión de volúmenes con libvirt

```
virsh vol-list default
virsh vol-list default --details
virsh vol-info prueba1.qcow2 default
virsh vol-dumpxml vol.qcow2 default

# Crear un volumen qcow2 de 10 GB en el pool default
virsh vol-create-as default vol1.qcow2 --format qcow2 10G

# Eliminar un volumen
virsh vol-delete vol1.qcow2 default
```

Gestión de volúmenes con `qemu-img`

Trabajamos con un pool de tipo `dir`:

```
cd /var/lib/libvirt/images

# Crear imagen qcow2 de 2 GB
qemu-img create -f qcow2 vol2.qcow2 2G

# Ver información de la imagen
qemu-img info vol2.qcow2

# Hacer que libvirt detecte el nuevo fichero
virsh pool-refresh vm-images
```

Creación de MV usando volúmenes existentes

```
virt-install \
             --virt-type kvm \
             --name prueba4 \
             --cdrom ~/iso/debian-11.3.0-amd64-netinst.iso \
             --os-variant debian10 \
             --disk vol=default/vol1.qcow2 \
             --memory 1024 \
             --vcpus 1
```

Otras formas de indicar el disco:

```
--disk path=/var/lib/libvirt/images/vol1.qcow2
--disk pool=vm-images,size=10
```

Añadir nuevos discos a una MV

```
# Añadir disco (también funciona en caliente)
virsh attach-disk prueba4 \
        /srv/images/vol2.qcow2 vdb \
        --driver=qemu --type disk \
        --subdriver qcow2 \
        --persistent

# Eliminar disco
virsh detach-disk prueba4 vdb --persistent
```

ℹ️ `--persistent` guarda el cambio en la configuración XML de la MV para que persista tras reinicios.

Redimensión de discos

Con la MV **parada**:

```
# Con libvirt
virsh vol-resize vol2.qcow2 3G --pool vm-images

# Con qemu-img
sudo qemu-img resize /srv/images/vol2.qcow2 3G
```

Con la MV **en ejecución** (en caliente):

```
virsh domblklist prueba4
virsh blockresize prueba4 /srv/images/vol2.qcow2 3G
```

Dentro de la MV, redimensionar el sistema de ficheros:

Redimensión del sistema de ficheros de una imagen

Para redimensionar el SF **sin entrar en la MV** usamos `virt-resize`:

```
# 1. Ampliar el fichero de imagen
qemu-img resize vol1.qcow2 10G

# 2. Copiar la imagen (virt-resize trabaja origen → destino)
cp vol1.qcow2 newvol1.qcow2

# 3. Expandir la partición dentro de la imagen
virt-resize --expand /dev/sda1 vol1.qcow2 newvol1.qcow2

# 4. Reemplazar la imagen original
mv newvol1.qcow2 vol1.qcow2
```

⚠️ `virt-resize` requiere el paquete `libguestfs-tools`. Siempre trabaja sobre una copia del fichero original.

IV · QEMU/KVM Y LIBVIRT

¡Gracias!

QEMU/KVM y libvirt

📧 José Domingo Muñoz 🏫 IES Gonzalo Nazareno · Dos Hermanas 📚 IV