

virt-lib Quick Reference

virt-top

This is not always installed by default. When run, it will bring up a full console window that shows all virtuals running on a system, similar to the top command.

virsh

The main controller for virt-lib. The basic format of the command is

```
virsh command parameters
```

. *man virsh* or simply *virsh -help* are your friends.

Some common commands are:

- **list -all** - list all domains virt-lib knows about and show their state (running, off, whatever)
- **start *name*** - starts the domain named **name** (name from list -all)
- **reboot *name***
- **shutdown *name***
- **dumpxml *name*** dumps the config to STDOUT
- **list -autostart** - list all domains which will autostart
- **autostart *domain*** - set domain to be autostarted in the future
- **autostart -disable *domain*** - unset autostart flag for domain
- **console *domain*** - attaches to the serial console of domain assuming the port has been set up.
- Attach/Detach network interfaces
(<https://computingforgeeks.com/managing-kvm-network-interfaces-in-linux/>)
 - `attach-interface -domain pxe -type bridge -source br1 -model virtio -config -live`
 - `detach-interface -domain pxe -type bridge -mac 52:54:00:47:2f:eb -config`

Remove a network from the entire system

```
virsh net-list --all  
virsh net-destroy br1  
virsh net-undefine br1  
virsh net-list --all
```

convert config file to native

```
virsh -c xen:/// domxml-to-native --format xen-xl \  
/path/to/libvirt/vm.domxml.cfg > vm.xenxl.cfg
```

Upgrading a disk to virtio (Unix)

I messed up and did not use virtio as the bus for one of my setups, which resulted in horrible disk I/O. I could have rebuilt the virtual (it was a simple install and I had not gone very far), but I decided to learn how to do it the “right way”. The trick here is to detach the disk, then re-attach it with the proper parameters.

Warning Linux (and FreeBSD) have the virtio drivers built in, but Microsoft products do not. Read the section on Microsoft products if you are trying to do this with Windows.

In my case, I wanted to reconfigure the boot drive, so I needed the machine down. It was running under the scsi bus, and I wanted virtio. Note that this is using LVM2 as the back end.

```
# detach the first drive (sda) from the domain router
virsh detach-disk router sda --config
# reattach it using the virtio bus.
virsh attach-disk router /dev/vg/lvname sda --config --targetbus=virtio
```

Normally, you attach/detach disks from a running system. The `--config` parameter allows you to do it on an inactive domain; it just erases the entry, then rebuilds it.

Upgrading a disk to virtio (Windows)

For Windows, you need to have the correct device drivers installed before you change the disk. This assumes the domain is running

1. Get a copy of the Windows VirtIO-win in iso onto your machine, someplace virsh can access, and mount the ISO
 1. Download ISO

```
cd /media/virtstorage && wget
https://fedorapeople.org/groups/virt/virtio-win/direct-
downloads/stable-virtio/virtio-win.iso
```

2. Mount it on a running system

```
virsh attach-disk domainname /media/virtstorage/stable-
virtio/virtio-win.iso hdc --type cdrom --mode readonly
```

2. Create a temporary disk and attach it. Use virtio for the format of it.

1. `virsh attach-disk domainname /path/to/disk/image sdb --targetbus=virtio`

3. Open your console to the windows domain

1. Find the new disk (sdb). It will have a yellow triangle, indicating it can not be read because there is no device driver. Right click and say “Update Driver”, then go find the drivers from the CD.

2. Shut down your Windows guest
4. Detach, then attach the primary drive.

```
# detach the first drive (sda) from the domain router
virsh detach-disk domainname sda --config
# reattach it using the virtio bus.
virsh attach-disk domainname /dev/vg/lvname sda --config --
targetbus=virtio
```

5. Start the Windows guest back up. Since the drivers are already installed, it should come back up.

I did this with a Windows 10 installation on my workstation. Prior to using virtio, it would take, literally, 5-7 minutes after boot before I could do anything, and it was very sluggish after that. Once I used virtio, it was almost bare hardware speeds.

Links

- <https://www.caretech.io/2017/10/18/convert-windows-vm-hard-disk-interface-to-virtio-with-oxmox-ve-5/>
- <https://docs.fedoraproject.org/en-US/quick-docs/creating-windows-virtual-machines-using-virtio-drivers/>
- https://linuxhint.com/install_virtio_drivers_kvm_qemu_windows_vm/

From:
<https://kb.unixservertech.com/> - **Unix Server Tech Knowledge Base**

Permanent link:
<https://kb.unixservertech.com/unix/virtualization/virtlib/quickreference?rev=1604131499>

Last update: **2020/10/31 08:04**

