Qubes OS Cheatsheet

Qubes Cheatsheet

a summary of useful qubes commands

version: 2.1

VM Management

qvm-block - *list/set VM PCI devices*

usage:

- `qvm-block -l [options]`
- `qvm-block -a [options] <device> <vm-name>`
- `qvm-block -d [options] <device>`
- `qvm-block -d [options] <vm-name>`

`qvm-block -A personal dom0:/home/user/extradisks/data.img` - *attaches an additional storage for the personal-vm*

qvm-clone - *clones an existing VM by copying all its disk files*

usage: `qvm-clone [options] <existing-vm-name> <new-clone-vm-name>`

`qvm-clone fedora-23 fedora-23-dev` - *create a clone of fedora-21 called fedora-21-dev*

qvm-firewall - *manage VM's firewall rules*

usage: `qvm-firewall -l [-n] <vm-name>`

`qvm-firewall -l personal` - *displays the firewall settings for the personal-vm*

`qvm-firewall -l -n fedora-23` - *displays the firewall settings for the personal-vm with port numbers*

qvm-ls - *list VMs and various information about their state*

usage: `qvm-ls [options] <vm-name>`

`qvm-ls` - *lists all vms*

`qvm-ls -n` - *show network addresses assigned to VMs*

`qvm-ls -d` - *show VM disk utilization statistics*

qvm-prefs - *list/set various per-VM properties*

usage:

- `qvm-prefs -l [options] <vm-name>`
- `qvm-prefs -s [options] <vm-name> <property> [...]`

`qvm-prefs win7-copy` - *lists the preferences of the win7-copy*

`qvm-prefs win7-copy -s mac 00:16:3E:5E:6C:05` - *sets a new mac for the network card*

`qvm-prefs lab-win7 -s qrexec_installed true` - *sets the qrexec to installed*

`qvm-prefs lab-win7 -s qrexec_timeout 120` - *usefull for windows hvm based vms*

`qvm-prefs lab-win7 -s default_user joanna` - *sets the login user to joanna*

qvm-run - *runs a specific command on a vm*

usage: `qvm-run [options] [<vm-name>] [<cmd>]`

`qvm-run personal xterm` - *runs xterm on personal*

`qvm-run personal xterm --pass-io` - *runs xterm and passes all sdtin/stdout/stderr to the terminal*

`qvm-run personal "sudo dnf update" --pass-io --nogui` - *pass a specific command directly to the VM*

qvm-start - *starts a vm*

usage: `qvm-start [options] <vm-name>`

`qvm-start personal` - *starts the personal-vm*

`qvm-start ubuntu --cdrom personal:/home/user/Downloads/ubuntu-14.04.iso` - *starts the ubuntu-vm with the ubuntu installation CD*

qvm-shutdown - *shutdowns a vm*

usage: `qvm-shutdown [options] <vm-name>`

`qvm-shutdown personal` - *shutdowns the personal-vm*

`qvm-shutdown --all` - *shutdowns all VM's*

qvm-kill - *kills a VM - same as pulling out the power cord - immediate shutdown*

usage: `qvm-kill [options] <vm-name>`

`qvm-kill personal` - *pull the power cord for the personal-vm - immediate shutdown*

qvm-sync-appmenus - *updates desktop file templates for given StandaloneVM or TemplateVM*

usage: `qvm-sync-appmenus [options] <vm-name>`

`qvm-sync-appmenus archlinux-template` - *useful for custom .desktop files or distributions not using dnf*

Dom0

qubes-dom0-update - *updates software in dom0*

usage: `qubes-dom0-update [--clean][--check-only][--gui] [<yum opts>][<pkg list>]`

`sudo qubes-dom0-update` - *updates dom0*

`sudo qubes-dom0-update qubes-windows-tools` - *install the windows tools*

`sudo qubes-dom0-update kernel-3.19*` - *install the official Fedora kernel-3.19* with Xen support*

qubes-hcl-report - *generates a report about the hardware information*

usage: `qubes-hcl-report [<vm-name>]`

`qubes-hcl-report` - *prints the hardware information on the console (terminal)*

`qubes-hcl-report personal` - *sends the hardware information to the personal-vm under `/home/user`*

virsh - *management user tool for libvirt (hypervisor abstraction)*

usage: `virsh -c xen:/// <command> [<vm-name>]`

Example

Why? Connect if GUI/qrexec does not work for any reason. This way you can restart/investigate a failed service.

- In Dom0 terminal: `virsh -c xen:/// console personal`
- username: **root** without a password

(and when #1130 would be implmented the same for "user")

xl - *Xen management tool, based on LibXenlight*

usage: `xl <subcommand> [<args>]`

`xl dmesg` - *Dom0 dmesg output (first place to look for warning or error messages)*

`xl top` - *Monitor host and domains in realtime*

DomU

qvm-copy-to-vm - Copy file from one VM to another VM

usage: `qvm-copy-to-vm <vm-name> <file> [<file+>]` - *file* can be a single file or a folder

`qvm-copy-to-vm work Documents` - *copy the `Documents` folder to the work VM*

`qvm-copy-to-vm personal text.txt` - *copy the `text.txt` file to the personal VM*

Example

- Open a terminal in AppVM A (e. g. your personal vm)
- Let's assume we want to copy the `Documents` folder to AppVM B (e. g. your work VM)
- The command would be: `qvm-copy-to-vm work Documents`

DomU and Dom0

List installed qubes packages **Fedora**

In VM or Dom0: `rpm -qa \*qubes-\*` - *list (qubes-) installed packages*

Copy from & to Dom0

Dom0 -> VM

Qubes 3.1+ - *Windows + Linux*

`dom0` console: `qvm-copy-to-vm <vm-name> <file> [<file+>]` - *file* can be a single file or a folder

Example:

```
qvm-copy-to-vm personal screenshot-qubes-gui.png
```

The file will be in the `personal` VM in the `/home/user/QubesIncoming/dom0` folder

Qubes < 3.1 - *Linux only*

```
cat /path/to/file_in_dom0 |
 qvm-run --pass-io <dst_domain>
  'cat > /path/to/file_name_in_appvm'
```

Example:

```
@dom0 Pictures]$ cat my-screenshot.png |
qvm-run --pass-io personal
'cat > /home/user/my-screenshot.png'
```

VM -> Dom0

```
qvm-run --pass-io <src_domain>
 'cat /path/to/file_in_src_domain' >
  /path/to/file_name_in_dom0
```

Copy text between VM A and B

On VM A (source):

1. `CTRL+C`
2. `CTRL+SHIFT+C`

On VM B (destination):

3. `CTRL+SHIFT+V`
4. `CTRL+V`

Grow disk

qvm-grow-private - *increase private storage capacity of a specified VM*

usage: `qvm-grow-private <vm-name> <size>`

Example

- In dom0 konsole: `qvm-grow-private personal 40GB`
- In the personal VM: `sudo resize2fs /dev/xvdb`

AppVMs and TMPFS

Enlarge /tmp if you run out of space on the default ~200MB

`sudo mount -o remount,size=1024M /tmp` - *enlarge the space to 1024MB*

Inter VM Networking

- *Does not expose services to the outside world!*

Make sure:

- Both VMs are connected to the same firewall VM
- Qubes IP addresses are assigned to both VMs
- Both VMs are started

In Firewall VM terminal:

```
$ sudo iptables -I FORWARD 2 -s <IP address of A> -d <IP address of B> -j ACCEPT
```

- The connection will be unidirectional `A -> B`
- Optional: Bidirectional `A <-> B`

In Firewall VM terminal:

```
$ sudo iptables -I FORWARD 2 -s <IP address of B> -d <IP address of A> -j ACCEPT
```

- Check your settings (e. g. using ping)
- Persist your settings:

```
Assume:
  IP of A: 10.137.2.10
  IP of B: 10.137.2.11
```

In Firewall VM terminal:

```
$ sudo bash
# echo "iptables -I FORWARD 2 -s 10.137.2.10 -d 10.137.2.11 -j ACCEPT" >> /rw/config/qubes_firewall_user_script
#chmod +x /rw/config/qubes_firewall_user_script
```

for bidirectional access:

```
# echo "iptables -I FORWARD 2 -s 10.137.2.10 -d 10.137.2.11 -j ACCEPT" >> /rw/config/qubes_firewall_user_script
```

Add USB Wifi card to sys-net VM * - *attach a USB Wifi card to sys-net VM*

The bus and device number can be different than shown in this example:

1. `qvm-pci -l sys-net` - *list all attached pci devices of sys-net*
2. `lsusb` - *e. g.* **Bus 003** *Device 003: ID 148f:2870 Ralink Technology, Corp. RT2870 Wireless Adapter*
3. `readlink /sys/bus/usb/devices/003` - *Important Bus 003 -> 003*
4. The result of readlink: `../../../devices/pci-0/pci0000:00/0000:00:12.2/usb3` - *Important 00:12.2*
5. `qvm-pci -a sys-net 00:12.2` - *attach USB device 00:12.2 to sys-net*
6. `qvm-pci -l sys-ne` - *check if device 00:12.2* is

Templates

Fedora - *Fedora template specific*

Updating, Searching & Installing Packages

Fedora > 21

- installing packages: `dnf install <package-name>`
- search for a package: `dnf search <package-or-word>`
- updating template: `dnf update`

Fedora <= 21

- installing packages: `yum install <package-name>`
- search for a package: `yum search <package-or-word>`
- updating template: `yum update`

Repositories

NOTE: Does not work anymore under fedora 23

Repositories: `Start Menu >> Template:Fedora 21 >> Package Sources >> Enable third party repositories`

`Start Menu >> Template:Fedora 21 >> Package Sources >> Enable RPMFusion` - ENABLE RPMFusion, (already covers RPMFusion signing keys)

Fedora Minimal - *Fedora minimal template*

`sudo qubes-dom0-update qubes-template-fedora-21-minimal` - *installs the fedora-21-minimal template*

Debian - *Debian templates*

Installing the Template

- `sudo qubes-dom0-update qubes-template-debian-7` - *Debian 7 “Wheezy”*
- `sudo qubes-dom0-update qubes-template-debian-8` - *Debian 8 “Jessie”*

Updating, Searching & Installing Packages

- installing packages: `apt-get install <package-name>`
- search for a package: `apt-cache search <package-or-word>`
- updating template:
    1. `apt-get update`
    2. `apt-get dist-upgrade`

Qubes OS + Whonix - *Whonix is an debian based OS focused on anonymity, privacy and security*

Whonix has two parts:

1. Whonix-Gateway (uses TOR for all connections to the outside world)
2. Whonix-Workstation (for application)

Install Whonix

Whonix-Gateway TemplateVM Binary Install @Dom0:

```
sudo qubes-dom0-update --enablerepo=qubes-templates-community qubes-template-whonix-gw-experimental
```

Whonix-Workstation TemplateVM Binary Install @Dom0:

1. `export UPDATES_MAX_BYTES=$[ 4 * 1024 ** 3 ]`
2. `sudo qubes-dom0-update --enablerepo=qubes-templates-community qubes-template-whonix-ws`

Next Steps

1. Create a Whonix-gateway ProxyVM, through Qubes VM Manager
2. Create a Whonix-workstation AppVM, through Qubes VM Manager
3. Update your Whonix-Gateway and Whonix-Workstation TemplateVMs (how to -> see debian)
4. (Re)Start Whonix-Gateway ProxyVM
5. Start Whonix-Workstation AppVM

Archlinux Minimal - *Archlinux minimal template*

Installing the Template

1. In a VM:

   ```
   wget http://olivier.medoc.free.fr/rpm/noarch/
   qubes-template-archlinux-minimal-3.0.3-201507281153.noarch.rpm
   ```

2. Copy RPM-Package to Dom0

3. In Dom0: `sudo rpm -i qubes-template-archlinux-minimal-3.0.3-201507281153.noarch.rpm`

Updating, Searching & Installing Packages

- installing packages: `pacman -S <package-name> [<package-name-2>...<package-name-n>]`
- search for a package: `pacman -Ss <package-or-word>`
- updating template: `pacman -Syyu`

Create VM from VMware or VirtualBox images

1. Download the image in an AppVM
2. Install qemu-img tools - *e. g. `dnf install qemu-img` for fedora*
3. Convert the image to a raw format:
    - VMware: `qemu-img convert ReactOS.vmdk -O raw reactos.img`
    - VirtualBox: `qemu-img convert ReactOS.vdi -O raw reactos.img`