Contents

1. CREATE A GAMING HVM

Note: This version may not be the most up to date. The latest version can usually be found on the Qubes OS forum.

1.1 Hardware

To have an 'HVM' for gaming, you must have

- A dedicated GPU. By dedicated, it means: it is a secondary GPU, not the GPU used to display dom0. In 2023, 'Nvidia' and 'Amd' GPU work. Not tested with Intel GPUs. External GPU using thunderbolt work (Create a Gaming HVM
- A screen available for the gaming 'HVM'. (It can be a physical monitor or just to have multiple cables connected to the screen and switching between input source)
- Dedicated gaming mouse and keyboard.
- A lot of patience. GPU passthrough is not trivial, and you will need to spend time debugging.

1.2 IOMMU Group

You need to check what are the things/devices that are in the same IOMMU group as the GPU you want to passthrough. You can't see your IOMMU Group when you are using Xen (the information is hidden from dom0). So, start a live linux distribution, enable iommu in the grub options (iommu=1 iommu_amd=on), and then displayed the folder structure of /sys/kernel/iommu_group

```
#!/bin/bash
shopt -s nullglob
for g in /sys/kernel/iommu_groups/*; do
 echo "IOMMU Group ${g##*/}:"
 for d in $g/devices/*; do
  echo -e "\t$(lspci -nns ${d##*/})"
done
done
```

1.3 GRUB modification

You must hide your secondary GPU from dom0. To do that, you have to modify the GRUB. In a dom0 Terminal, type:

```
qvm-pci
```

Then find the devices id for your secondary GPU. In my case, it is `dom0:0a_00.0`. Edit `/etc/default/grub`, and add the PCI hiding.

```
GRUB_CMDLINE_LINUX="... rd.qubes.hide_pci=0a:00.0 "
```

then regenerate the grub

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

If you are using UEFI, the file to override with `grub2-mkconfig` is `/boot/efi/EFI/qubes/grub.cfg`.

Note: if after this step when you reboot the computer you get stuck in the QubesOS startup that means you are trying to use the GPU you just hide. Check your BIOS options. Also check the cables, BIOS have some GPU priority based on the type of cable. For example, DisplayPort can be favoured over HDMI.

Once you have rebooted, in dom0, type `sudo lspci -vvn`, you should see "Kernel driver in use: pciback" for the GPU you just hide.

1.4 Configuring the parameter "max-ram-below-4g"

Since the release of this qubes version of xen: 4.17.2-8 7 (R4.2, 2024-01-03), no additional configuration is required.

Remove any existing "max-ram-below-4g" workaround

1.5 Preparing the guest

As of 2023, I recommend using a Linux guest instead of a window guest.

1.5.1 Windows

Install a window VM, you can use this qvm-create-windows-qube

1.5.2 Linux

Create a new standalone Qube based on the template of your choice.

You must run the kernel provided by the guest distribution, because we will use some non-default kernel module for the GPU driver. Just follow the doc: managing-vm-kernel.

Install the GPU drivers you need.

1.6 Pass the GPU

In qubes settings for the HVM, go to the 'devices' tab, pass the ID corresponding to your GPU.

You may or may not need to add the option "permissive" or "no-strict-reset".

Some word about the security implication of thoses parameters.

```
qvm-pci attach gpu_gaming_archlinux dom0:0a_00.0 -o permissive=True
↪  -o no-strict-reset=True
```

1.7 Starting the guest

This is where you will have a lot of issues to debug.

For Linux guests, run 'sudo dmesg' to have all the kernel log indicating you if there is a issue with your GPU driver. For some hardware, the MSI calls won't work. You can work around that using for example `pci=nomsi` or `NVreg_EnableMSI=0` or something else. Check your drivers options. Check if alternative drivers exist (amdgpu, nvidia, nouveau, nvidia-open, using drivers from the official website, ...). Check multiple kernel version.

Some links that could help you to debug the issues you will have

- https://forum.qubes-os.org/t/ryzen-7000-serie/
- https://dri.freedesktop.org/docs/drm/gpu/amdgpu.html

For windows guests you will probably have the same issues but it will be harder to debug. I recommend using the drivers from Windows Update instead of the official drivers from the website of the constructor.

Some things that may be useful for debugging:

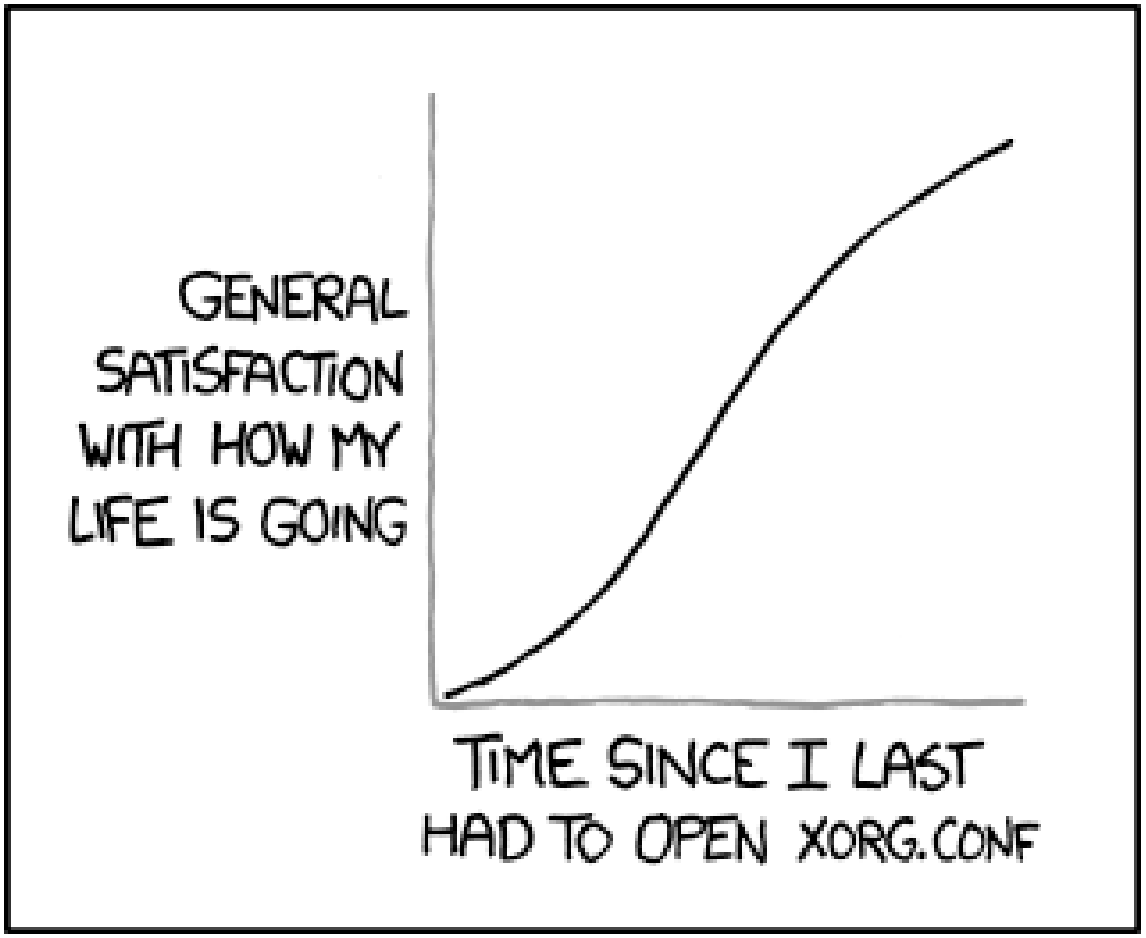
- Virsh (start, define, ...)
- /etc/libvirt/libxl/
- xl
- /etc/qubes/templates/libvirt/xen/by-name/
- /usr/lib/xen/boot/
- virsh -c xen:/// domxml-to-native xen-xm /etc/libvirt/libxl/...

Issues with the drivers could be related to 'qubes-vmm-xen-stubdom-linux', 'qubes-vmm-xen', and the Linux kernel you will be using.

1.8 Linux guest — Integration with QubesOS

1.8.1 Xorg

Now Xorg and Pulseaudio. From XKCD:



Things you need to install:

- The Xorg input driver to support your mouse and keyboard
- Your favorite Windows Manager

In my case, it is:

Archlinux version

```
pacman -S i3 xorg
```

Debian version

```
apt install xserver-xorg-input-kbd xserver-xorg-input-libinput
↪  xserver-xorg-input-mouse i3
```

Then create a XORG configuration file for your GPU and screen. My file named 'AOC.conf':

```
Section "ServerLayout"
Identifier "Gaming"
Screen 0 "AMD AOC" Absolute 0 0
EndSection

Section "Device"
```

```
Identifier  "AMD"

# name of the driver to use. Can be "amdgpu", "nvidia", or
↪  something else
Driver      "amdgpu"

# The BusID value will change after each qube reboot.
BusID       "PCI:0:8:0"
EndSection

Section "Monitor"
Identifier "AOC"
VertRefresh 60
# https://arachnoid.com/modelines/ .  IMPORTANT TO GET RIGHT. MUST
↪  ADJUST WITH EACH SCREEN.
Modeline "1920x1080" 172.80 1920 2040 2248 2576 1080 1081 1084 1118
EndSection

Section "Screen"
Identifier "AMD AOC"
Device     "AMD"
Monitor    "AOC"
EndSection
```

We can't know what is the correct BusID before the qube is started. And it change after each reboot. So let's write a script — named "xorgX1.sh" — that update this configuration file with the correct value, then start a binary on the Xorg X screen n°1.

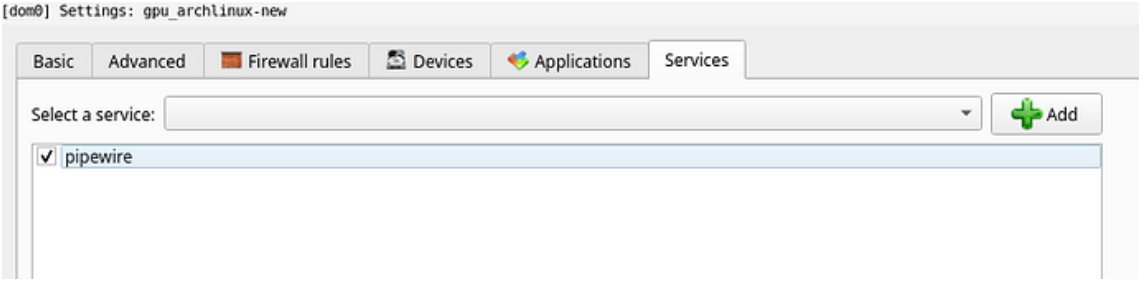
```
#!/bin/bash

binary=${1:?binary required}

# Find the correct BusID of the AMD GPU, then set it in the Xorg
↪  configuration file
pci=$(lspci | grep "VGA" | grep -E "NVIDIA|AMD/ATI" | cut -d " " -f
↪  1 | cut -d ":" -f 2 | cut -d "." -f 1 | cut -d "0" -f 2)
sed -i 's/"PCI:[^"]*"/"PCI:0:'$pci':0"/g' /home/user/AOC.conf

# Start the Xorg server for the X screen number 1.
# The X screen n°0 is already used for QubesOS integration
sudo startx "$binary" -- :1 -config /home/user/AOC.conf
```

1.8.2 Audio

Delete any packages related to pulseaudio and install pipewire

```
sudo pacman -Rdd qubes-vm-pulseaudio pulseaudio
sudo pacman -S pipewire-{jack,alsa,pulse} pipewire-qubes
```

Enable the “pipewire” service for this qube using Qubes Manager



Create a script to launch your favorite Windows Manager and force it to use a specific pulse server. Example “i3.sh”:

```
#!/bin/bash
sleep 5 && sudo setxkbmap -display :1 fr &
/bin/sudo -u user PULSE_SERVER=unix:/run/user/1000/pulse/native bash
↪  -c 'sudo xhost + local:;/usr/bin/i3'
```

And launch it:

```
./xorgX1.sh ./i3.sh
```

1.9 Contributors

- @neowutran
- @deeplow
- @xvrhthxn