Qubes OS

Gaming HVM and Nitrokey

Gaming HVM and Nitrokey

Author:
Neowutran

Contents

1 Create a Gaming Windows HVM

Setting up a Windows HVM for gaming in Qubes OS is not particulary hard - If you buy the right hardware and follow the docs -

1.1 References

Everythings needed is referenced here

- Usefull technical details
- Reddit thread of what is needed for GPU passthrough
- Solution to have more than 3Go of RAM in the Windows HVM
- Some old references

1.2 Prerequise

You have a functional Windows 7 HVM. The ”how to” for this part can be found on the Qubes OS documentation and here: Usefull github comment.

However, few tips:

- Do a backup (clone VM) of the Windows HVM BEFORE starting to install QWT
- The Windows user MUST BE ”user”
- Windows 7 Only, do not use Windows 10 or others.

1.3 Hardware

To have a Windows HVM for gaming, you must have:

- A dedicated AMD GPU. By dedicated, it means: it is a secondary GPU, not the GPU used to display dom0. Nvidia GPU are not supported (or maybe with a lot of trick).
- A really fast disk (M.2 disk)
- A lot of RAM
- A dedicated screen

In my case:

- Secondary GPU: AMD RX580
- Primary GPU: Some Nvidia trash, used for dom0
- 32Go of RAM. 16Go of RAM will be dedicated for the Windows HVM
- A fast M.2 disk

1.4 Checklist

Short list of things to do to make the GPU passthrough work:

- In dom0, you edited the file /etc/default/grub to allow PCI hidding for your secondary GPU, and regenerated the grub
- You patched your stubdom-linux-rootfs.gz to allow to have more than 3Go of RAM for your Windows HVM

1.5 GRUB modification

You must hide your secondary GPU from dom0. To do that, you have to edit 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" and "dom0:0a_00.1".

Edit /etc/default/grub, and add the PCI hiding

```
GRUB\_CMDLINE\_LINUX="....
rd.qubes.hide\_pci=0a:00.0,0a:00.1
 modprobe=xen-pciback.passthrough=1
 xen-pciback.permissive"
```

then regenerate the grub

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

1.6 Patching stubdom-linux-rootfs.gz

Follow the instructions here: https://github.com/QubesOS/qubes-issues/issues/4321#issuecomment-423011787

Copy-paste of the comment:

This is caused by the default TOLUD (Top of Low Usable DRAM) of 3.75G provided by qemu not being large enough to accommodate the larger BARs that a graphics card typically has. The code to pass a custom max-ram-below-4g value to the qemu command line does exist in the libxl_dm.c file of xen, but there is no functionality in libvirt to add this parameter. It is possible to manually add this parameter to the qemu commandline by doing the following in a dom0 terminal:

```
mkdir stubroot
cp /usr/lib/xen/boot/stubdom-linux-rootfs stubroot/stubdom-linux-rootfs.gz
cd stubroot
gunzip stubdom-linux-rootfs.gz
cpio -i -d -H newc --no-absolute-filenames < stubdom-linux-rootfs
rm stubdom-linux-rootfs
nano init
```

Before the line "# $dm_args and $kernel are separated with x1b to allow for spaces in arguments." add:

```
SP=$'\x1b'
dm_args=$(echo "$dm_args" \
 | sed "s/-machine\\${SP}xenfv/-machine\
\\${SP}xenfv,max-ram-below-4g=3.5G/g")
```

Then execute:

```
find . -print0 | cpio --null -ov \
--format=newc | gzip -9 > ../stubdom-linux-rootfs
sudo mv ../stubdom-linux-rootfs /usr/lib/xen/boot/
```

Note that this will apply the change to all HVMs, so if you have any other HVM with more than 3.5G ram assigned, they will not start without the adapter being passed through. Ideally to fix this libvirt should be extended to pass the max-ram-below-4g parameter through to xen, and then a calculation added to determine the correct TOLUD based on the total BAR size of the PCI devices are being passed through to the vm.

1.7 Pass the GPU

In qubes settings for the windows HVM, go to the "devices" tab, pass the ID corresponding to your AMD GPU. (in my case, it was 0a:00.0 and 0a:00.1) And check the option for "nostrict reset" for those 2.

1.8 Conclusion

Don't forget to install the GPU drivers, you can install the official one from AMD website, no modification or trick to do. Nothing else is required to make it work (in my case at least, once I finish to fight to find those informations). If you have issues, you can refer to the links in the first sections. If it doesn't work and you need to debug more things, you can go deeper.

- 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/...

I am able to play games on my windows HVM with very good performances. And safely.

1.9 Bugs

The AMD GPUs have a bug when used in HVM: each time you will reboot your windows HVM, it will get slower and slower. It is because the AMD GPUs is not correctly resetted when you restart your windows HVM Two solutions for that:

- Reboot your computer
- In the windows HVM, use to windows option in the system tray to "safely remove devices", remove your GPU. Restart the HVM.

This bug is referenced somewhere, but lost the link and too lazy to search for it.

2 Nitrokey and QubeOS

To use a nitrokey on QubeOS, a USB passhrough is required. This means, you need to have a sys-usb VM. This is mentioned in the Qubes Documentation.

```
Note, you cannot pass through devices from dom0
(in other words: a USB VM is required).
```

If you are using a USB keyboard, the sys-usb VM is not installed by default. If you are using a USB keyboard, you have 2 options:

- Create a sys-usb VM and assign a USB Controller to it.
- If you can't assign a USB Controller (ex: You only have 1 on your computer and can't buy another), then buy and use a PS/2 Keyboard.