Fedora 30 QEMU-KVM OVMF Passthrough

My Hardware

Motherboard: Z370 AORUS Gaming 5 (rev. 1.0)
CPU: Intel(R) Core(TM) i7-8700K CPU
RAM: 64 GB CORSAIR Vengeance LPX 2666
GPU: RTX 2080, GTX 1050
PSU: EVGA SuperNOVA 850 G3
STORAGE: 2 HDD's, 1 SSD, 2 NVME

Packages to install

```
sudo dnf install virt qemu kvm qemu-img libvirt virt-install
sudo usermod -a -G libvirt username
sudo systemctl enable libvirtd
```

Configuring Host before passing through

> **Make sure you do not have the GPU you want to passthrough in your slot #0 of your PCI lanes. This will alter the ROM as soon as the host is booted, and you will be unable to use your GPU properly on your guest.**

The script below will show you all PCI devices and their mapping to their respective IOMMU groups. If the output is blank, you do not have IOMMU enabled.

```
#!/bin/bash
shopt -s nullglob
for d in /sys/kernel/iommu_groups/*/devices/*; do
    n=${d#*/iommu_groups/*}; n=${n%%/*}
    printf 'IOMMU Group %s ' "$n"
    lspci -nns "${d##*/}"
done;
```

Enabling IOMMU :

You will need to add a boot load kernel option :

```
vim /etc/sysconfig/grub
```

Add: *rd.driver.pre=vfio-pci i915.alpha_support=1 intel_iommu=on iommu=pt* at the end of your *GRUB_CMDLINE_LINUX=*

The grub config will look like this:

```
GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="resume=UUID=90cb68a7-0260-4e60-ad10-d2468f4f6464 rhgb quiet
rd.driver.pre=vfio-pci i915.alpha_support=1 intel_iommu=on iommu=pt"
GRUB_DISABLE_RECOVERY="true"
```

Re-gen your grub2

```
grub2-mkconfig -o /boot/efi/EFI/fedora/grub.cfg
```

```
reboot
```

Use script above // Example output :

```
IOMMU Group 0 00:00.0 Host bridge [0600]: Intel Corporation 8th Gen Core Processor Host
Bridge/DRAM Registers [8086:3ec2] (rev 07)
IOMMU Group 10 00:1c.2 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root
Port #3 [8086:a292] (rev f0)
IOMMU Group 11 00:1c.3 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root
Port #4 [8086:a293] (rev f0)
IOMMU Group 12 00:1c.4 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root
Port #5 [8086:a294] (rev f0)
IOMMU Group 13 00:1c.6 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root
Port #7 [8086:a296] (rev f0)
IOMMU Group 14 00:1d.0 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root
Port #9 [8086:a298] (rev f0)
IOMMU Group 15 00:1f.0 ISA bridge [0601]: Intel Corporation Z370 Chipset LPC/eSPI Controller
[8086:a2c9]
IOMMU Group 15 00:1f.2 Memory controller [0580]: Intel Corporation 200 Series/Z370 Chipset
```

```
Family Power Management Controller [8086:a2a1]
IOMMU Group 15 00:1f.3 Audio device [0403]: Intel Corporation 200 Series PCH HD Audio [8086:a2f0]
IOMMU Group 15 00:1f.4 SMBus [0c05]: Intel Corporation 200 Series/Z370 Chipset Family SMBus Controller [8086:a2a3]
IOMMU Group 16 00:1f.6 Ethernet controller [0200]: Intel Corporation Ethernet Connection (2) I219-V [8086:15b8]
IOMMU Group 17 02:00.0 Non-Volatile memory controller [0108]: Sandisk Corp WD Black NVMe SSD [15b7:5001]
IOMMU Group 18 07:00.0 USB controller [0c03]: ASMedia Technology Inc. Device [1b21:2142]
IOMMU Group 19 08:00.0 Network controller [0280]: Intel Corporation Wireless 3165 [8086:3165] (rev 81)
IOMMU Group 1 00:01.0 PCI bridge [0604]: Intel Corporation Xeon E3-1200 v5/E3-1500 v5/6th Gen Core Processor PCIe Controller (x16) [8086:1901] (rev 07)
IOMMU Group 1 01:00.0 VGA compatible controller [0300]: NVIDIA Corporation TU104 [GeForce RTX 2080] [10de:1e87] (rev a1)
IOMMU Group 1 01:00.1 Audio device [0403]: NVIDIA Corporation Device [10de:10f8] (rev a1)
IOMMU Group 1 01:00.2 USB controller [0c03]: NVIDIA Corporation Device [10de:1ad8] (rev a1)
IOMMU Group 1 01:00.3 Serial bus controller [0c80]: NVIDIA Corporation Device [10de:1ad9] (rev a1)
IOMMU Group 2 00:02.0 VGA compatible controller [0300]: Intel Corporation UHD Graphics 630 (Desktop) [8086:3e92]
IOMMU Group 3 00:08.0 System peripheral [0880]: Intel Corporation Xeon E3-1200 v5/v6 / E3-1500 v5 / 6th/7th Gen Core Processor Gaussian Mixture Model [8086:1911]
IOMMU Group 4 00:14.0 USB controller [0c03]: Intel Corporation 200 Series/Z370 Chipset Family USB 3.0 xHCI Controller [8086:a2af]
IOMMU Group 5 00:16.0 Communication controller [0780]: Intel Corporation 200 Series PCH CSME HECI #1 [8086:a2ba]
IOMMU Group 6 00:17.0 SATA controller [0106]: Intel Corporation 200 Series PCH SATA controller [AHCI mode] [8086:a282]
IOMMU Group 7 00:1b.0 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root Port #17 [8086:a2e7] (rev f0)
IOMMU Group 8 00:1b.4 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root Port #21 [8086:a2eb] (rev f0)
IOMMU Group 9 00:1c.0 PCI bridge [0604]: Intel Corporation 200 Series PCH PCI Express Root Port #1 [8086:a290] (rev f0)
```

Isolating your GPU

To assign a GPU device to a Virtual machine, you will need to use a place holder driver to prevent the host from interacting with it on boot. You cannot dynamically re-assign a GPU device on a VM after you booted due to its complexity. You can use either VFIO or pci-stub.

Most newer machines will have VFIO by default, which we will be using here.

If your system supports it, which you can try by running the following command, you should use it. If it returns an error, use pci-stub instead.

```
modinfo vfio-pci
-----
filename:       /lib/modules/4.9.53-1-lts/kernel/drivers/vfio/pci/vfio-pci.ko.gz
description:    VFIO PCI - User Level meta-driver
author:         Alex Williamson <alex.williamson@redhat.com>
license:        GPL v2
```

In this case here I'm interested in the following groups to passthrough

```
IOMMU Group 1 01:00.0 VGA compatible controller [0300]: NVIDIA Corporation TU104 [GeForce RTX 2080] [10de:1e87] (rev a1)
IOMMU Group 1 01:00.1 Audio device [0403]: NVIDIA Corporation Device [10de:10f8] (rev a1)
IOMMU Group 1 01:00.2 USB controller [0c03]: NVIDIA Corporation Device [10de:1ad8] (rev a1)
IOMMU Group 1 01:00.3 Serial bus controller [0c80]: NVIDIA Corporation Device [10de:1ad9] (rev a1)
```

Adding their relevant IDs to the VFIO driver

> **After you completed the below steps, your GPU will no longer be detected by your host, make sure you have a secondary GPU available.**

```
vim /etc/modprobe/vfio.conf
```

```
options vfio-pci ids=10de:1e87,0de:10f8,0de:1ad8,0de:1ad9
```

Regenerate initramfs

```
dracut -f --kver `uname -r`
```

```
reboot
```

```
lsmod | grep vfio
----
vfio_pci               53248  5
irqbypass              16384  11 vfio_pci,kvm
vfio_virqfd            16384  1 vfio_pci
vfio_iommu_type1       28672  1
vfio                   32768  10 vfio_iommu_type1,vfio_pci
```

Create a Bridge

```
sudo nmcli connection add type bridge autoconnect yes con-name br0 ifname br0
sudo nmcli connection modify br0 ipv4.addresses 10.1.2.120/24 ipv4.method manual
sudo nmcli connection modify br0 ipv4.gateway 10.1.2.10
sudo nmcli connection modify br0 ipv4.dns 10.1.2.10
sudo nmcli connection del eno1
sudo nmcli connection add type bridge-slave autoconnect yes con-name eno1 ifname eno1 master
br0
```

Remove current interface from boot

```
vim /etc/sysconfig/network-scripts/ifcfg-Wired_connection_1
```

```
ONBOOT=no
```

```
vim /tmp/br0.xml
```

```
virsh net-define /tmp/br0.xml
virsh net-start br0
virsh net-autostart br0
virsh net-list --all
```

```
reboot
```

Create KVM VM

Example of XML file from my VM

```
<domain
type='kvm'>









  <name>win10-nvme</name>
  <uuid>7f99dec1-f092-499e-92f8-bd2d2fab8a5c</uuid>
  <memory unit='KiB'>18524160</memory>
  <currentMemory unit='KiB'>18524160</currentMemory>
  <vcpu placement='static'>12</vcpu>
  <cputune>
    <vcpupin vcpu='0' cpuset='6'/>
    <vcpupin vcpu='1' cpuset='7'/>
    <vcpupin vcpu='2' cpuset='8'/>
    <vcpupin vcpu='3' cpuset='9'/>
    <vcpupin vcpu='4' cpuset='10'/>
    <vcpupin vcpu='5' cpuset='11'/>
  </cputune>
  <os>
    <type arch='x86_64' machine='pc-i440fx-2.11'>hvm</type>
    <loader readonly='yes' type='pflash'>/usr/share/edk2/ovmf/OVMF_CODE.fd</loader>
    <nvram>/usr/share/edk2/ovmf/OVMF_VARS.fd</nvram>
    <bootmenu enable='no'/>
  </os>
  <features>
    <acpi/>
    <apic/>
    <hyperv>
      <relaxed state='on'/>
      <vapic state='on'/>
      <spinlocks state='on' retries='8191'/>
      <vendor_id state='on' value='whatever'/>
```

```
  </hyperv>
  <kvm>
    <hidden state='on'/>
  </kvm>
  <vmport state='off'/>
</features>
<cpu mode='host-passthrough' check='none'>
  <topology sockets='1' cores='6' threads='2'/>
</cpu>
<clock offset='localtime'>
  <timer name='rtc' tickpolicy='catchup'/>
  <timer name='pit' tickpolicy='delay'/>
  <timer name='hpet' present='no'/>
  <timer name='hypervclock' present='yes'/>
</clock>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>destroy</on_crash>
<pm>
  <suspend-to-mem enabled='no'/>
  <suspend-to-disk enabled='no'/>
</pm>
<devices>
  <emulator>/usr/bin/qemu-system-x86_64</emulator>
  <disk type='block' device='disk'>
    <driver name='qemu' type='raw' cache='none' io='native'/>
    <source dev='/dev/sdb'/>
    <target dev='sdb' bus='sata'/>
    <boot order='1'/>
    <address type='drive' controller='0' bus='0' target='0' unit='2'/>
  </disk>
  <disk type='block' device='disk'>
    <driver name='qemu' type='raw' cache='none' io='native'/>
    <source dev='/dev/nvme0n1'/>
    <target dev='sdd' bus='sata'/>
    <boot order='2'/>
    <address type='drive' controller='0' bus='0' target='0' unit='0'/>
  </disk>
  <controller type='pci' index='0' model='pci-root'/>
```

```
<controller type='sata' index='0'>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x06' function='0x0'/>
</controller>
<controller type='virtio-serial' index='0'>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x07' function='0x0'/>
</controller>
<controller type='usb' index='0' model='nec-xhci'>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'/>
</controller>
<interface type='bridge'>
  <mac address='52:54:00:4b:a0:2a'/>
  <source bridge='br0'/>
  <model type='rtl8139'/>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x03' function='0x0'/>
</interface>
<input type='mouse' bus='ps2'/>
<input type='keyboard' bus='ps2'/>
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0000' bus='0x00' slot='0x14' function='0x0'/>
  </source>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x05' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0000' bus='0x01' slot='0x00' function='0x0'/>
  </source>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x04' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0000' bus='0x01' slot='0x00' function='0x1'/>
  </source>
  <address type='pci' domain='0x0000' bus='0x00' slot='0x08' function='0x0'/>
</hostdev>
<hostdev mode='subsystem' type='pci' managed='yes'>
  <source>
    <address domain='0x0000' bus='0x01' slot='0x00' function='0x2'/>
  </source>
```

```
      <address type='pci' domain='0x0000' bus='0x00' slot='0x09' function='0x0'/>
    </hostdev>
    <hostdev mode='subsystem' type='pci' managed='yes'>
      <source>
        <address domain='0x0000' bus='0x01' slot='0x00' function='0x3'/>
      </source>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0a' function='0x0'/>
    </hostdev>
    <memballoon model='virtio'>
      <address type='pci' domain='0x0000' bus='0x00' slot='0x0b' function='0x0'/>
    </memballoon>
  </devices>
</domain>
```

Extra Notes:

How i did my CPU pinning

```
grep -e "processor" -e "core id" -e "^$" /proc/cpuinfo
```

```
processor□: 0
core id□□: 0

processor□: 1
core id□□: 1

processor□: 2
core id□□: 2

processor□: 3
core id□□: 3

processor□: 4
```

```
core id□□: 4

processor□: 5
core id□□: 5

processor□: 6
core id□□: 0

processor□: 7
core id□□: 1

processor□: 8
core id□□: 2

processor□: 9
core id□□: 3

processor□: 10
core id□□: 4

processor□: 11
core id□□: 5
```

```
  <cputune>
    <vcpupin vcpu='0' cpuset='6'/>
    <vcpupin vcpu='1' cpuset='7'/>
    <vcpupin vcpu='2' cpuset='8'/>
    <vcpupin vcpu='3' cpuset='9'/>
    <vcpupin vcpu='4' cpuset='10'/>
    <vcpupin vcpu='5' cpuset='11'/>
  </cputune>
```

virtio drivers (only needed when installing windows in a qcow2)

https://docs.fedoraproject.org/en-US/quick-docs/creating-windows-virtual-machines-using-virtio-drivers/index.html

---

Revision #27
Created 2017-08-23 22:59:17 UTC by Carl
Updated 2019-07-01 22:34:32 UTC by Dave