Compute and GPU Passthough

This section is dedicated to compute related things, from Xen to GPU/vGPU or PCI passthrough

Compute and GPU

PCI Passthrough

#0. Prerequisites

WARNING

Ensure VT-d/IOMMU Support Is Enabled

In order to use PCI passthrough your host system must have VT-d/IOMMU functionality enabled. This should be more commonly enabled by default on enterprise hardware than on consumer hardware. It can be enabled in the BIOS/UEFI of systems with CPUs and chipsets which support it. For Intel platforms the feature is typically referred to as VT-d (Intel Virtualization Technology for Directed I/O); on AMD platforms it is typically listed as IOMMU or AMD-Vi. Please note that this is not the same as VT-x/AMD-v virtualisation support, and so these options are often listed separately.

Consult your system or motherboard manual for instructions on where to find the setting in your BIOS/UEFI. In addition, system BIOS updates may reset the feature to its default state, which may require you to re-enable it.

If you attempt to perform PCI passthrough on a system which does not have VT-d/IOMMU enabled, you may encounter the following error when you start the target virtual machine:

```
Internal error: xenopsd internal error: Device.PCI.Cannot_add(_, _)
```

WARNING

You may not be able to passthrough USB controllers

```
Internal error: xenopsd internal error: Cannot_add(0000:00:1d.0, Xenctrlext.Unix_error(30, "1:
Operation not permitted"))
```

and an error in `/var/log/xen/hypervisor.log`

```
[2020-08-22 10:09:03] (XEN) [  297.542134] [VT-D] It's disallowed to assign 0000:08:00.0 with shared RMRR at 7ba77000 for Dom32753.
[2020-08-22 10:09:03] (XEN) [  297.542136] d[IO]: assign (0000:08:00.0) failed (-1)
```

This indicates that your device is using RMRR (opens new window). Intel IOMMU does not allow DMA to these devices (opens new window)and therefore PCI passthrough is not supported.

#1. Find your devices ID (B/D/F (opens new window)) on the PCI bus using one of the following methods:

Method 1: List PCI Devices with `lspci`

This method is the easiest way to find devices.

```
[root@xen ~]# lspci
...
04:01.0 Ethernet controller: Intel Corporation 82541PI Gigabit Ethernet Controller (rev 05)
```

Method 2: List System Device Classes with `find`

This method works best for finding the device ID by class. The example below the class is `net` and shows how to find the device ID of a specific network interface.

```
[root@xen ~]# find /sys/class/net -exec readlink {} +
../../devices/virtual/net/lo
../../devices/pci0000:00/0000:04:01.0/net/eth1
```

#2. Tell XCP-ng not to use this device ID for Dom0

Add the **`xen-pciback.hide`** parameter to the kernel boot parameters:

```
/opt/xensource/libexec/xen-cmdline --set-dom0 "xen-pciback.hide=(0000:04:01.0)"
```

> You can hide multiple devices. If you wanted to add another device at `00:19.0` just append it to the parameter.
>
> `/opt/xensource/libexec/xen-cmdline --set-dom0 "xen-pciback.hide=(0000:04:01.0)(0000:00:19.0)"`

To remove any passthrough devices from dom0:

```
/opt/xensource/libexec/xen-cmdline --delete-dom0 xen-pciback.hide
```

> **TIP**
>
> This kernel parameter is not retained when you upgrade an XCP-ng host using the installation ISO. Remember to re-do this step after the upgrade.

#3. Reboot the XCP-ng host

`[root@xen ~]# reboot`

#4. Check with `xl pci-assignable-list` on CLI

```
[root@xen ~]# xl pci-assignable-list
0000:04:01.0
```

#5. Put this PCI device 'into' your VM

`[root@xen ~]# xe vm-param-set`**`other-config:pci=0/0000:04:01.0`**`uuid=<vm uuid>`

> You can also pass through multiple devices. If you wanted to pass through another device at `00:19.0` just append it to the parameter.
>
> `[root@xen ~]# xe vm-param-set`**`other-config:pci=0/0000:04:01.0,0/0000:00:19.0`**
> `uuid=<vm uuid>`

#6. Start your VM and be happy ?

`[root@xen ~]# xe vm-start uuid=<vm uuid>`

#GPU Passthrough

To passthrough a complete graphics card to a VM (not virtualize it into multiple virtual vGPUs, which is different, see the vGPU section below), just follow the regular PCI passthrough instructions, no special steps are needed. Most Nvidia and AMD video cards should work without issue.

> **TIP**
>
> Previously, Nvidia would block the use of gaming/consumer video cards for passthrough (the Nvidia installer would throw an **Error 43** when installing the driver inside your VM). They lifted this restriction in 2021 with driver R465 and above, so be sure to use the latest driver. Details from Nvidia here.(opens new window)

#vGPU

#NVIDIA vGPU

WARNING

Due to a proprietary piece of code in XenServer, XCP-ng doesn't have (yet) support for NVIDIA vGPUs.

#MxGPU (AMD vGPU)

AMD GPU are trivial using industry standard.
Version 2.0 of the mxgpu iso should work on any 8.X version of XCP-ng

1. Enable SR-IOV in the server's BIOS
2. Install XCP-ng
3. Download Citrix XenServer from AMD's Drivers & Support page. (Currently version 2.0.0 for XenServer 8.1)
4. Copy the `mxgpu-2.0.0.amd.iso` to the host
5. Install the supplemental pack:

```
cd /tmp
xe-install-supplemental-pack mxgpu-2.0.0.amd.iso
```

1. Reboot the XCP-ng
2. Assign an MxGPU to the VM from the VM properties page. Go to the GPU section. From the Drop down choose how big of a slice of the GPU you want on the VM and click OK

Start the VM and log into the guest OS and load the appropriate guest driver from AMD's Drivers & Support page.

> Known working cards:

- S7150x2

#USB Passthrough

TIP

There's no need to alter any files manually as some older guides suggest

It's fairly easy using the `xe` CLI. First use `xe pusb-list` to list all the physical USB devices on your host available for passthrough:

```
[root@xenserver ~]# xe pusb-list
uuid ( RO)            : 10fbec89-4472-c215-5d55-17969b473ee6
             path ( RO): 2-1.1
       vendor-id ( RO): 0781
     vendor-desc ( RO): SanDisk Corp.
      product-id ( RO): 5591
    product-desc ( RO):
          serial ( RO): 4C530001151223117134
         version ( RO): 2.10
     description ( RO): SanDisk Corp._4C530001151223117134
```

Find your USB device there, and note the `uuid`. Then use that uuid to enable passthrough for it:

```
[root@xenserver ~]# xe pusb-param-set uuid=10fbec89-4472-c215-5d55-17969b473ee6 passthrough-enabled=true
```

This will create a `usb-group` containing this USB device. We need to find the uuid of that group, so we use the `usb-group-list` command, specifying the physical USB uuid we got in step one:

```
[root@xenserver ~]# xe usb-group-list PUSB-uuids=10fbec89-4472-c215-5d55-17969b473ee6
uuid ( RO)                : 1f731f6a-6025-8858-904d-c98548f8bb23
name-label ( RW): Group of 0781 5591 USBs
name-description ( RW):
```

Note the uuid of this usb-group, then use it in the following command to attach this USB device to your desired VM. Remember to first shut down the target VM as hot-plug for USB passthrough is not supported:

```
xe vusb-create usb-group-uuid=<usb_group_uuid> vm-uuid=<vm_uuid>
```

So using the examples above, it would look like:

```
xe vusb-create usb-group-uuid=1f731f6a-6025-8858-904d-c98548f8bb23 vm-uuid=4feeb9b2-2176-b69d-b8a8-cf7289780a3f
```

Finally, start the target guest VM:

```
[root@xenserver ~]# xe vm-start uuid=<vm_uuid>
```

Note: If you get a message containing `internal error` when trying to start the VM after assigning it a USB device, try the following command to ensure its `platform:usb` parameter is set correctly:

```
xe vm-param-set uuid=<vm_uuid> platform:usb=True
```

In the future if you ever need to unplug the virtual USB device from your VM, or remove and unassign it completely, find the uuid of the virtual USB device by running `xe vusb-list`. Then use the uuid of the virtual USB device in one or both of the following commands:

```
xe vusb-unplug uuid=<vusb_uuid>
xe vusb-destroy uuid=<vusb_uuid>
```

#Passing through Keyboards and Mice

xcp-ng host uses usb-policy.conf at `/etc/xensource/usb-policy.conf` with ALLOW and DENY rules for different classes of usb devices. The default file contains Mice and Keyboards with DENY rules. You can edit this file to allow these devices (and any other ones similarly).

Once edited, run the following command to refresh:

```
/opt/xensource/libexec/usb_scan.py -d
```

Then run

```
xe pusb-scan host-uuid=<host_uuid>
```