LXC

- LXC GPU Access
- LXC NIC Passthrough
- netfilter/iptable logging
- LXC USB Passthrough

LXC GPU Access

Giving a LXC guest GPU access allows you to use a GPU in a guest while it is still available for use in the host machine. This is a big advantage over virtual machines where only a single host or guest can have access to a GPU at one time. Even better, multiple LXC guests can share a GPU with the host at the same time.

> The information on this page is written for a host running Proxmox but should be easy to adapt to any machine running LXC/LXD.

> Since a device is being shared between two systems there are almost certainly some security implications and I haven't been able to determine what degree of security you're giving up to share a GPU.

Determine Device Major/Minor Numbers

To allow a container access to the device you'll have to know the devices major/minor numbers. This can be found easily enough by running `ls -l` in `/dev/`. As an example to pass through the integated UHD 630 GPU from an Core i7 8700k you would first list the devices where are created under `/dev/dri`.

```
root@blackbox:~# ls -l /dev/dri
total 0
drwxr-xr-x 2 root root        80 May 12 21:54 by-path
crw-rw---- 1 root video  226,   0 May 12 21:54 card0
crw-rw---- 1 root render 226, 128 May 12 21:54 renderD128
```

From that you can see the major device number is `226` and the minors are `0` and `128`.

Provide LXC Access

In the configuration file you'd then add lines to allow the LXC guest access to that device and then also bind mount the devices from the host into the guest. In the example above since both devices share the same major number it is possible to use a shorthand notation of `226:*` to represent all minor numbers with major number `226`.

```
# /etc/pve/lxc/*.conf
+ lxc.cgroup.devices.allow: c 226:* rwm
```

```
+ lxc.mount.entry: /dev/dri/card0 dev/dri/card0 none bind,optional,create=file,mode=0666
+ lxc.mount.entry: /dev/dri/renderD128 dev/dri/renderD128 none bind,optional,create=file
```

Allow `unprivileged` Containers Access

In the example above we saw that `card0` and `renderD128` are both owned by `root` and have their groups set to `video` and `render`. Because the "unprivilged" part of LXC unprivileged container works by mapping the UIDs (user IDs) and GIDs (group IDs) in the LXC guest namespace to an unused range of IDs on host, it is necessary to create a custom mapping for that namespace that maps those groups in the LXC guest namespace to the host groups while leaving the rest unchanged so you don't lose the added security of running an unprivilged container.

First you need to give root permission to map the group IDs. You can look in `/etc/group` to find the GIDs of those groups, but in this example `video` = `44` and `render` = `108` on our host system. You should add the following lines that allow `root` to map those groups to a new GID.

```
# /etc/subgid
+ root:44:1
+ root:108:1
```

Then you'll need to create the ID mappings. Since you're just dealing with group mappings the UID mapping can be performed in a single line as shown on the first line addition below. It can be read as "remap `65,536` of the LXC guest namespace UIDs from `0` through `65,536` to a range in the host starting at `100,000`." You can tell this relates to UIDs because of the `u` denoting users. It wasn't necessary to edit `/etc/subuid` because that file already gives root permission to perform this mapping.

You have to do the same thing for groups which is the same concept but slightly more verbose. In this example when looking at `/etc/group` in the LXC guest it shows that `video` and `render` have GIDs of `44` and `106`. Although you'll use `g` to denote GIDs everything else is the same except it is necessary to ensure the custom mappings cover the whole range of GIDs so it requires more lines. The only tricky part is the second to last line that shows mapping the LXC guest namespace GID for `render` (`106`) to the host GID for `render` (`108`) because the groups have different GIDs.

```
# /etc/pve/lxc/*.conf
  lxc.cgroup.devices.allow: c 226:* rwm
  lxc.mount.entry: /dev/dri/card0 dev/dri/card0 none bind,optional,create=file,mode=0666
  lxc.mount.entry: /dev/dri/renderD128 dev/dri/renderD128 none bind,optional,create=file
+ lxc.idmap: u 0 100000 65536
+ lxc.idmap: g 0 100000 44
+ lxc.idmap: g 44 44 1
+ lxc.idmap: g 45 100045 61
+ lxc.idmap: g 106 108 1
```

```
+ lxc.idmap: g 107 100107 65429
```

Beaues it can get confusing to read I just wanted show each line with some comments...

```
+ lxc.idmap: u 0 100000 65536    // map UIDs 0-65536 (LXC namespace) to 100000-165535 (host namespace)
+ lxc.idmap: g 0 100000 44       // map GIDs 0-43 (LXC namspace) to 100000-100043 (host namespace)
+ lxc.idmap: g 44 44 1           // map GID  44 to be the same in both namespaces
+ lxc.idmap: g 45 100045 61      // map GIDs 45-105 (LXC namspace) to 100045-100105 (host namespace)
+ lxc.idmap: g 106 108 1         // map GID  106 (LXC namspace) to 108 (host namespace)
+ lxc.idmap: g 107 100107 65429  // map GIDs 107-65536 (LXC namspace) to 100107-165536 (host namespace)
```

Add `root` to Groups

Because `root`'s UID and GID in the LXC guest's namespace isn't mapped to `root` on the host you'll have to add any users in the LXC guest to the groups `video` and `render` to have access the devices. As an example to give `root` in our LXC guest's namespace access to the devices you would simply add `root` to the `video` and `render` group.

```
usermod --append --groups video,render root
```

Potential Alernative

lxc.mount.entry - static uid/gid in LXC guest

Resources

Proxmox: Unprivileged LXC containers

LXC NIC Passthrough

On the rare occation you have a good reason to forgo the small overhead of an *veth* (Virtual Ethernet) device connected to an ethernet bridge it is possible to pass a physical network interface directly to a LXC host.

To pass a physical device you just need to provide `lxc.net.[index].type` and `lxc.net.[index].link` parameters in the LXC config. You may optionally provide a name for the link as well with `lxc.net.[index].name`. Just be sure your index value is unique among all network interfaces fot the LXC container including those Proxmox may add if you running your LXC hosts on Proxmox.

```
lxc.net.0.type: phys
lxc.net.0.link: enp1s0
# optional
lxc.net.0.name: eth0
```

netfilter/iptable logging

> Logging from network namespaces other than init has been disabled since kernel 3.10 in order to prevent host kernel log flooding from inside a container.

Source: lxc-users.linuxcontainers.narkive.com

There are two ways to get logging working on guests running in Namespaces. The first is to simply enable it on even though it is off by default due to the security concerns mentioned above. The second *and better* way is to use User space logging which doesn't carry the same restrictions because it doesn't interact with Kernel space in the same way. Besides the User space logging method being the best security practice, anytime it is possible to modify the host machine less it is better in my opinion.

Method 1: Userspace Logging (on guest)

Install `ulogd2`

```
apt install ulogd2
```

Replace `LOG` in any `iptable/netfilter` rules with `NFLOG`

```
- -A INPUT -j LOG
+ -A INPUT -j NFLOG
```

Source: lxadm.com

Method 2: Enable Logging In Namespaces (on host)

> Logging from network namespaces other than init has been disabled since kernel 3.10 in order to prevent host kernel log flooding from inside a container.
>
> If you have kernel >= 4.11 or one with commit 2851940ffee3 ("netfilter: allow logging from non-init namespaces") backported, you can enable netfilter logging from other network namespaces by...

```
sysctl net.netfilter.nf_log_all_netns=1
```

Source: lxc-users.linuxcontainers.narkive.com

This will enable all netfilter (the `nf` part in `nf_log_all_netns`) logging from namespaces until the next reboot. It can also be enabled persistently using one of the following methods...

Option 1: Always On with `sysctl.conf`

Add a single line to `sysctl.conf` so the setting gets applied at boot.

```
echo "net.netfilter.nf_log_all_netns = 1" >> /etc/sysctl.conf
```

Option 2: On Demand with Snippets (for Proxmox only)

Add a bash script to use as a `snippet`.

```
# /var/lib/vz/snippets/nf_log_all_netns.sh
+ #!/bin/bash
+
+ case $2 in
+   pre-start)
+     echo "[pre-start]"
+     echo -e "\tEnabling netfilter namespace logging."
+     echo -e "\t$(sysctl net.netfilter.nf_log_all_netns=1)"
+     ;;
+   pre-stop)
+     echo "[pre-stop]"
+     echo -e "\tDisabling netfilter namespace logging."
+     echo -e "\t$(sysctl net.netfilter.nf_log_all_netns=0)"
+     ;;
+ esac
```

Then add the "*hookscript*" to that container. If your container ID was `100` it would look like

```
$ pct set 100 -hookscript local:snippets/nf_log_all_netns.sh
```

LXC USB Passthrough

Passing through a USB device with LXC allows your LXC guest access to a physical USB device plugged into the host system.

> The information on this page is written for a host running Proxmox but should be easy to adapt to any machine running LXC/LXD.

Locate Bus/Device

```
root@vault:~# lsusb
Bus 002 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 001 Device 003: ID 13d3:3273 IMC Networks 802.11 n/g/b Wireless LAN USB Mini-Card
Bus 001 Device 004: ID 10c4:8a2a Silicon Labs HubZ Smart Home Controller
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Determine Device Major/Minor Numbers

```
root@vault:~# ls -l /dev/bus/usb/001/004
crw-rw-r-- 1 root root 189, 3 Oct  3 17:17 /dev/bus/usb/001/004
```

From that you can see the major device number is `189` and the minor is `3`.

Provide LXC Access

In the configuration file you'd then add lines to allow the LXC guest access to that device and then also bind mount the devices from the host into the guest. In the example above since both devices share the same major number it is possible to use a shorthand notation of `189:*` to represent all minor numbers with major number `189`.

```
# /etc/pve/lxc/*.conf
+ lxc.cgroup.devices.allow: c 189:* rwm
+ lxc.mount.entry: /dev/bus/usb/001/020 dev/bus/usb/001/020 none bind,optional,create=file,mode=664
```

Allow `unprivileged` Containers Access

> incomplete

Resources

USB Passthrough to an LXC (Proxmox)