Si en el host de PVE tenemos una tarjeta gráfica que queremos exponer hacia un contenedor LXC no-privilegiado, podemos hacer lo siguiente:

Primero determinamos que tipo de tarjeta gráfica tiene el host:

```
lspci | grep -E "VGA|3D|Display"
```

En mi caso:

```
15:00.0 VGA compatible controller: NVIDIA Corporation AD106 [GeForce RTX 4060 Ti] (rev a1)
16:00.0 VGA compatible controller: Advanced Micro Devices, Inc. [AMD/ATI] Baffin [Radeon RX 460/560D / Pro
450/455/460/555/555X/560/560X] (rev e5)
```

Lo siguiente es encontrar que dispositivo de render corresponde a la tarjeta que queremos pasar. Por ejemplo, si lo que queremos es pasar la RX 560, hacemos

```
ls -l /dev/dri/
```

En mi caso, con esta salida:

```
total 0
drwxr-xr-x  2 root root   80       nov  9 01:16 by-path
crw-rw----+ 1 root video  226, 1   nov 10 14:17 card1
crw-rw----+ 1 root render 226, 128 nov  9 01:16 renderD128
```

Y os preguntaréis, como aparece sólo un dispositivo de render cuando hay dos tarjetas gráficas. Bueno, puede ser que sólo una tarjeta gráfica tenga capacidades 3D y, como lo que ocurre en mi caso, la RTX4060Ti se pasa por PCI-Passthrough a una máquina virtual por lo que el host pierde su dispositivo de render asociado.

Entonces, como en este caso el dispositivo de Render es único (renderD128), las cosas se nos ponen más fáciles.

¿Qué pasaría si no fuese único? Pues que tendríamos que averiguar su PATH:

```
udevadm info -q all -n /dev/dri/renderD128 | grep ID_PATH
```

Posible salida:

```
E: ID_PATH=pci-0000:16:00.0
E: ID_PATH_TAG=pci-0000_16_00_0
```

Vemos que 0000:16:00.0 si que corresponde con el path de la tarjeta gráfica RX 560 que nos salió cuando ejecutamos lspci, por lo que confirmamos que todo está correcto antes de exponer el render al contenedor. Entonces, para exponerlo, añadimos al archivo de configuración del contenedor (por ejemplo /etc/pve/lxc/102.conf) las siguientes líneas:

```
unprivileged: 1
lxc.cgroup2.devices.allow: c 226:0 rwm
lxc.cgroup2.devices.allow: c 226:128 rwm
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 62
lxc.idmap: g 107 104 1
```

```
lxc.idmap: g 108 100108 65428
```

Con esto, el contenedor no-privilegiado tendrá acceso al render node de la GPU, lo que permite a aplicaciones dentro del LXC usar aceleración por hardware (por ejemplo, con Mesa, VAAPI o CUDA, según el caso).

Pero, con el propósito educativo de entender cada una de esas líneas de configuración, vamos a proceder a explicarlas:

unprivileged: 1

Esto indica que el contenedor es no privilegiado. En este modo, el usuario root dentro del contenedor no es root real del host, sino un UID mapeado (por ejemplo, UID 100000 en el host). Esto evita que el contenedor pueda manipular directamente dispositivos del sistema o afectar al host. Es la base de la seguridad de los LXC no privilegiados.

lxc.cgroup2.devices.allow: c 226:0 rwm

Permite al contenedor acceder a un dispositivo de tipo carácter (c) con número mayor 226 y menor 0. En este caso, 226 corresponde al subsystem “DRI” (Direct Rendering Infrastructure), que usan las tarjetas gráficas. El menor 0 corresponde normalmente a /dev/dri/card0. El sufijo rwm indica que el contenedor tiene permiso de lectura (r), escritura (w) y manejo de ioctl (m) sobre ese dispositivo.

lxc.cgroup2.devices.allow: c 226:128 rwm

Permite el acceso al dispositivo /dev/dri/renderD128, que es el render node de la GPU. A diferencia de /dev/dri/card0, este nodo no da acceso al control del dispositivo, solo al renderizado por software o hardware, lo cual es más seguro y suficiente para tareas gráficas y cómputo (como OpenCL, VAAPI, etc.).

lxc.mount.entry: /dev/dri/renderD128 dev/dri/renderD128 none bind,optional,create=file

Monta el archivo de dispositivo /dev/dri/renderD128 del host dentro del contenedor, en la misma ruta /dev/dri/renderD128.

- bind: significa que se monta el archivo tal cual desde el host.
- optional: evita que el arranque falle si el dispositivo no existe.
- create=file: crea un archivo vacío si no está presente, para mantener la estructura /dev/dri dentro del contenedor.

Así, cuando una aplicación dentro del LXC accede a /dev/dri/renderD128, en realidad está accediendo al nodo real del host.

lxc.idmap: u 0 100000 65536

Define el mapa de UIDs (usuarios). Dentro del contenedor, el UID 0 (root) se mapea al UID 100000 del host, y hay un rango de 65536 UIDs consecutivos disponibles. Esto es lo que hace que el contenedor sea “no privilegiado”: root dentro del contenedor no es root fuera.

lxc.idmap: g 0 100000 44
lxc.idmap: g 44 44 1
lxc.idmap: g 45 100045 62
lxc.idmap: g 107 104 1
lxc.idmap: g 108 100108 65428

Estas líneas hacen lo mismo que la anterior, pero para GIDs (grupos). El mapeo de grupos es más fino porque permite exponer al contenedor ciertos grupos del host, lo cual es necesario para que tenga acceso al dispositivo gráfico.

- lxc.idmap: g 44 44 1: El grupo 44 dentro del contenedor (normalmente video) se mapea al mismo GID 44 en el host. Esto es clave, porque los nodos /dev/dri/* en el host pertenecen al grupo video. Así, los procesos dentro del LXC que pertenezcan a video podrán acceder al render node.
- Las demás líneas amplían el rango de grupos para mantener compatibilidad con los GIDs de sistema del contenedor (por ejemplo, de 0 a 100000) y otros servicios (104, 108, etc.) que podrían necesitarse.

Todas estas líneas de configuración combinadas permiten que el contenedor no privilegiado acceda al render node de la GPU sin darle privilegios directos sobre el dispositivo físico y garantizando que los permisos de grupo video se mantengan coherentes entre host y contenedor.