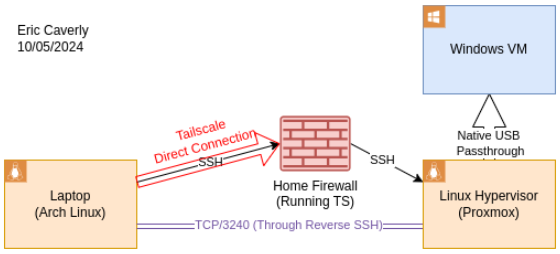
USBIP

Networking Overview



This guide does not include any configuration for

- Proxmox
- Tailscale
- Firewalls
- Windows
- Remote Desktop Protocol
- SSH Keys

This was to keep it as general as possible. Please research how to set these up on your own if you are not familiar with them.

Installation

> **Summary**
>
> In my setup, I am "forwarding" a device connected to my laptop (running arch-linux) to a remote device (proxmox hypervisor) over a tailscale VPN connection.
>
> Therefore, the server is the arch-linux laptop

Server (Laptop)

Install and enable the service

```
sudo pacman -S usbip

sudo systemctl enable usbipd
```

> **Hint**
>
> usbip requires kernel modules to be installed.

Enable kernel modules

```
sudo modprobe usbip_core
sudo modprobe usbip_host
```

> In Arch I found these to be added to `/etc/modules` automatically

Verify the service is running

```
sudo systemctl start usbipd

sudo systemctl status usbipd
```

```
# Expected output:
● usbipd.service - USB/IP server
```

```
     Loaded: loaded (/usr/lib/systemd/system/usbipd.service; enabled; preset: disabled)
     Active: active (running) since Fri 2024-10-04 18:33:33 MDT; 32min ago
 Invocation: 44416ae393c94d5ebf948ec9dfa6a7ab
   Main PID: 834 (usbipd)
      Tasks: 1 (limit: 18716)
     Memory: 636K (peak: 1.5M)
        CPU: 69ms
     CGroup: /system.slice/usbipd.service
             └─834 /usr/bin/usbipd
```

Client (Proxmox)

Install USBIP

```
apt update
apt install usbip
```

Manually load and set auto load of kernel modules

```
modprobe usbip-core && echo 'usbip-core' >> /etc/modules
modproble vhci-hcd && echo 'vhci-hcd' >> /etc/modules
```

Run script on Laptop

> **Tip**
>
> Replace the ID variable contents with the ID of YOUR device found with `lsusb`

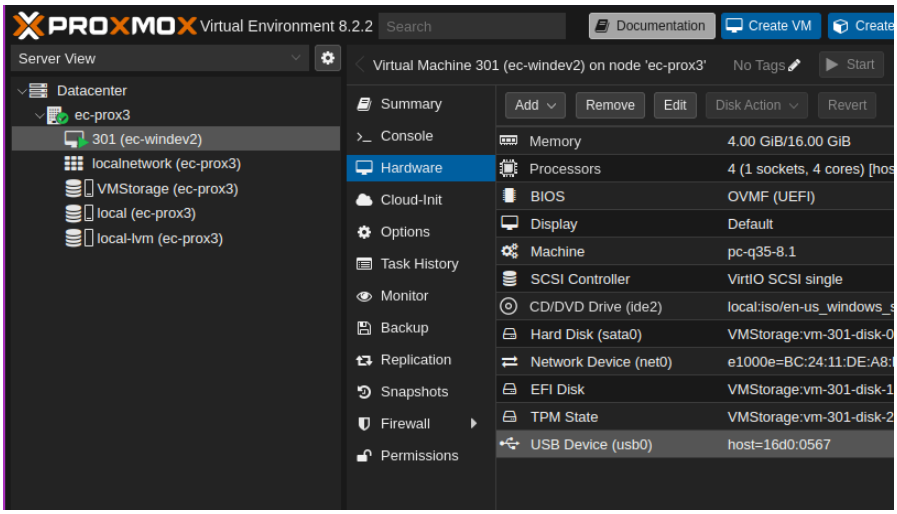
```
#!/bin/bash

# USB ID of device to forward
ID="16d0:0567"

# Server (Actually USBIP Client) Information
USR="root"
REMOTE="10.0.10.40"

# Find the USBIP Bus of the device
BUS=$(sudo usbip list -pl | grep $ID | sed -n 's/.*busid=\([^#]*\).*/\1/p' )

echo -e "---[LOCAL]---"
# Inform the user of the BUS
echo -e "Forwarding USB BUS $BUS"

# Bind the bus to this computer
sudo usbip bind -b $BUS

# Remote into the remote computer, while forwarding the local USBIP server (TCP/3240) to the remote server (which can then access it on
127.0.0.1:3240)
# Immediately run the command to attach the USB device on the BUS found before available at localhost (which ends up being the port
forwarded from this computer)
# Wait 1 second for the connection to be formed. Show a connected message, ports connected, and inform the user how to exit
echo -e "---[SERVER]---"
ssh -R localhost:3240:127.0.0.3:3240 $USR@$REMOTE "usbip attach -r 127.0.0.1 -b $BUS & sleep 1; usbip port; echo 'connected,CTRL+C to exit'"

echo -e "---[LOCAL]---"
# Unbind the USB port from this computer
sudo usbip unbind -b $BUS
```

This script does the following:

- Finds the USB BUS that has the ID on it
- Binds the USB Bus
- Builds a reverse SSH tunnel to the proxmox server, forwarding the bound USB port
- Within the SSH session, runs the command to attach the USB device.
- Prints attached device information and waits for CTRL+C
- Upon CTRL+C being pressed, the SSH session is destroyed, and the port is unbound on the local computer

> **Hint**
>
> Because the port is being forwarded through an SSH tunnel, it does **NOT** need to be opened on your laptop firewall!

Pass through USB from Proxmox to VM



1. Log into the Proxmox Web GUI
2. Click on VM --> Hardware --> Add --> USB Device
3. Select "Use USB Vendor/Device ID"
4. Choose the forwarded device
5. Add

Now check in the VM if it is present.

> **Will this cause problems if USB is not forwarded?**
>
> No, according to Proxmox's own documentation this will not cause problems.

Conclusion

You should now have the USB device forwarded onto the VM through a reverse SSH tunnel. Congrats!