Homelab: Fully Managed Server

- PVE: NVIDIA GPU Passthrough
- PVE: Custom VLANs
- PBS: Backup Strategy

PVE: NVIDIA GPU Passthrough

This guide walks you through creating a Linux VM in Proxmox (with GPU pass-through or related capabilities) and installing the NVIDIA driver on both **Arch Linux** and **Debian** (Bookworm).

---

1. Proxmox VM Configuration

1. **Locate and edit the VM configuration file**:

```
/etc/pve/qemu-server/<VMID>.conf
```

2. **Add the following settings**:

```
args: -cpu 'host,+kvm_pv_unhalt,+kvm_pv_eoi,hv_vendor_id=NV43FIX,kvm=off'
bios: ovmf
machine: q35
vga: none
```

3. **Note about VGA**:
   - For the initial operating system installation, you may need to set:

     ```
     vga: std
     ```

   - After installation, change it to:

     ```
     vga: none
     ```

   and then reboot the VM.

---

2. Update GRUB

2.1 For Arch Linux

1. **Edit the GRUB configuration**:

```
sudo nano /etc/default/grub
```

2. **Modify the `GRUB_CMDLINE_LINUX_DEFAULT` line** to include:

```
GRUB_CMDLINE_LINUX_DEFAULT="loglevel=3 quiet splash nvidia-drm.modeset=1"
```

(Add any other parameters you need for your setup here.) 3. **Regenerate the GRUB configuration**:

```
sudo grub-mkconfig -o /boot/grub/grub.cfg
```

2.2 For Debian (Bookworm)

1. **Edit the GRUB configuration**:

```
sudo nano /etc/default/grub
```

2. **Modify or add these lines**:

```
GRUB_CMDLINE_LINUX_DEFAULT="quiet"
GRUB_CMDLINE_LINUX="nvidia-drm.modeset=1"
```

3. **Update GRUB**:

```
sudo update-grub
```

---

3. Update Package Sources

3.1 For Arch Linux

Arch Linux uses rolling releases, so you typically do **not** need to edit package sources. Simply keep your system up to date:

```
sudo pacman -Syu
```

3.2 For Debian (Bookworm)

1. **Edit the sources list**:

```
sudo nano /etc/apt/sources.list
```

2. **Ensure it contains something like**:

```
deb http://deb.debian.org/debian/ bookworm main non-free-firmware contrib non-free
deb-src http://deb.debian.org/debian/ bookworm main non-free-firmware contrib non-free
deb http://security.debian.org/debian-security bookworm-security main non-free-firmware
contrib non-free
deb-src http://security.debian.org/debian-security bookworm-security main non-free-firmware
contrib non-free
deb http://deb.debian.org/debian/ bookworm-updates main non-free-firmware contrib non-free
deb-src http://deb.debian.org/debian/ bookworm-updates main non-free-firmware contrib non-free
```

3. **Update and upgrade your packages**:

```
sudo apt update
sudo apt upgrade
```

---

4. Install the NVIDIA Driver

4.1 For Arch Linux

1. **Install the NVIDIA driver**:

```
sudo pacman -S nvidia
```

2. **Create/edit the modprobe configuration file**:

```
sudo nano /etc/modprobe.d/nvidia.conf
```

3. **Add the following lines**:

```
options nvidia_drm modeset=1
options nvidia_drm fbdev=1
```

4. **Save and close** the file.

4.2 For Debian (Bookworm)

1. **Install the NVIDIA driver**:

```
sudo apt install nvidia-driver
```

2. Depending on your hardware and kernel, Debian may automatically create the appropriate modprobe files. If additional configuration is needed, you can place them in `/etc/modprobe.d/`.

---

5. Secure Boot Configuration

Before rebooting, decide how you want to handle Secure Boot. If your VM or host uses UEFI with Secure Boot enabled, you have two main options:

Option A: Disable Secure Boot (Easier)

1. **Disable validation**:

```
mokutil --disable-validation
```

2. **Follow the on-screen prompts on the first reboot** to complete the process.

Option B: Sign the Driver (More Secure)

- Refer to your distribution's documentation on how to sign kernel modules:
  - Arch Linux NVIDIA Documentation
  - Debian NVIDIA Graphics Drivers
- This process involves creating or importing a Machine Owner Key (MOK), signing the NVIDIA kernel module, and enrolling the key with your firmware.

---

6. Reboot

After completing all the steps relevant to your distribution and Secure Boot configuration:

1. **Reboot your system**:

```
sudo reboot
```

2. **Verify that the NVIDIA drivers are in use** by checking:
   - **Arch Linux**:

```
nvidia-smi
```

- **Debian**:

```
nvidia-smi
```

If the command shows information about your NVIDIA GPU, the driver is successfully loaded.

Final Notes

- **VM Performance**: If you are using this VM for GPU pass-through, make sure your Proxmox host IOMMU groups and passthrough settings are properly configured.
- **Troubleshooting**:
  - Check kernel messages (e.g., `dmesg`) for signs of driver load failures.
  - Ensure your VM configuration (`qemu-server/<VMID>.conf`) properly masks the CPU vendor (`hv_vendor_id=NV43FIX`) if you are trying to hide the virtualization from the driver (useful in some GPU pass-through scenarios).

PVE: Custom VLANs

This guide presents two methods for setting up VLANs in Proxmox and configuring a UniFi switch to work with them.

Prerequisites

- Proxmox VE installed
- Root access to the Proxmox host
- UniFi Network Controller access
- Network interface(s) available for configuration

Method 1: Manual VLAN Creation Without Explicit VLAN Tagging

Proxmox Configuration

1. **Access the Proxmox host**
   - SSH into your Proxmox host or access the console directly
2. **Edit the network configuration file**
   - Open the network interfaces configuration file:

     ```
     nano /etc/network/interfaces
     ```

3. **Configure the main bridge (vmbr0) and VLAN bridge (vmbr1)**
   - Add the following configuration:

     ```
     auto vmbr0
     iface vmbr0 inet static
             address 192.168.1.7/24
             gateway 192.168.1.1
             bridge-ports eno1
             bridge-stp off
             bridge-fd 0

     auto vmbr1
     ```

```
iface vmbr1 inet static
        address 192.168.2.1/24
        bridge-ports eno1.2
        bridge-stp off
        bridge-fd 0
        bridge-vlan-aware yes
        bridge-vids 2-4094

source /etc/network/interfaces.d/*
```

4. **Save and apply the configuration**
   - Save the file and exit the editor
   - Restart networking or reboot the Proxmox host:

     ```
     systemctl restart networking
     ```

     or

     ```
     reboot
     ```

UniFi Switch Configuration for Method 1

1. **Access the UniFi Network Controller**
   - Log in to your UniFi Network Controller interface
2. **Navigate to the Devices section**
   - Find and select the UniFi switch connected to your Proxmox host
3. **Locate the correct port**
   - Identify the port number that your Proxmox host is connected to
4. **Configure the port for multiple VLANs**
   - Click on the port to open its configuration settings
   - Set the "Port Profile" to "All"
   - In the "Native VLAN" field, enter the VLAN ID for your main network (usually 1)
   - In the "Tagged VLANs" field, enter "2-4094" to allow all possible VLANs
5. **Enable VLAN awareness on the switch**
   - In the switch settings, ensure that "VLAN Aware" is turned on
6. **Create VLANs in UniFi Controller**
   - Go to the "Settings" > "Networks" section in your UniFi Controller
   - Create a new network for each VLAN you plan to use
   - Assign appropriate VLAN IDs to these networks (matching the ones you set up in Proxmox)
7. **Configure DHCP and routing (if needed)**
   - If you want the UniFi Controller to handle DHCP for your VLANs, configure DHCP servers for each VLAN network

- Set up appropriate firewall rules to control traffic between VLANs

8. **Apply the changes**
   - Save the port configuration
   - Apply the changes to the switch
9. **Verify the configuration**
   - Check the UniFi Controller's insights or statistics to ensure traffic is flowing correctly on the configured VLANs

Method 2: Using VLAN Tags in Proxmox VMs and UniFi

Proxmox Configuration

1. **Access the Proxmox host**
   - SSH into your Proxmox host or access the console directly
2. **Edit the network configuration file**
   - Open the network interfaces configuration file:

     ```
     nano /etc/network/interfaces
     ```

3. **Configure the main bridge (vmbr0)**
   - The main bridge typically does not need to be changed. Here's an example of a basic default configuration:

     ```
     auto lo
     iface lo inet loopback

     iface eno1 inet manual

     auto vmbr0
     iface vmbr0 inet static
             address 192.168.1.100/24
             gateway 192.168.1.1
             bridge-ports eno1
             bridge-stp off
             bridge-fd 0

     source /etc/network/interfaces.d/*
     ```

   - Adjust the `address` and `gateway` as needed for your network
4. **Save and apply the configuration**

- Save the file and exit the editor
- Restart networking:

```
systemctl restart networking
```

5. **Configure VLAN tagging for VMs**
   - When creating or editing a VM in the Proxmox web interface:
     - Go to the VM's "Hardware" tab
     - Add a new network device or edit an existing one
     - Set "Bridge" to vmbr0
     - In the "VLAN Tag" field, enter the desired VLAN ID (e.g., 10, 20, 30)

UniFi Switch Configuration for Method 2

1. **Access the UniFi Network Controller**
   - Log in to your UniFi Network Controller interface
2. **Navigate to the Devices section**
   - Find and select the UniFi switch connected to your Proxmox host
3. **Locate the correct port**
   - Identify the port number that your Proxmox host is connected to
4. **Configure the port for tagged VLANs**
   - Click on the port to open its configuration settings
   - Set the "Port Profile" to "All"
   - In the "Native VLAN" field, enter the VLAN ID for your main network (usually 1)
   - In the "Tagged VLANs" field, enter the VLAN IDs you plan to use in your Proxmox VMs (e.g., "10,20,30")
5. **Create VLANs in UniFi Controller**
   - Go to the "Settings" > "Networks" section in your UniFi Controller
   - Create new networks for each VLAN, matching the IDs you plan to use in Proxmox VMs
6. **Configure DHCP and routing (if needed)**
   - If you want the UniFi Controller to handle DHCP for your VLANs, configure DHCP servers for each VLAN network
   - Set up appropriate firewall rules to control traffic between VLANs
7. **Apply the changes**
   - Save the port configuration
   - Apply the changes to the switch
8. **Verify the configuration**
   - Check the UniFi Controller's insights or statistics to ensure traffic is flowing correctly on the configured VLANs

Comparison of Methods

- **Method 1** uses a VLAN-aware bridge in Proxmox, which can be more flexible for the host system but may be more complex to set up initially.
- **Method 2** keeps the Proxmox network configuration simple and uses VLAN tagging at the VM level. This method is more straightforward and aligns directly with how most network equipment handles VLANs.

Choose the method that best fits your network architecture and management preferences. Method 2 is often preferred for its simplicity and flexibility in managing VLANs on a per-VM basis.

Troubleshooting

- Verify VLAN IDs match between Proxmox (either in the host configuration for Method 1 or VM settings for Method 2) and UniFi configurations
- Check UniFi firewall rules for inter-VLAN traffic
- Use UniFi Controller's built-in tools to test connectivity between VLANs
- In Proxmox, use these commands to verify VLAN configurations:

  ```
  ip a
  bridge vlan show
  ```

- For Method 2, ensure the VLAN tag is correctly set in each VM's network device settings
- If using Method 1, check that the VLAN-aware bridge (vmbr1) is correctly configured and up
- Test connectivity from within VMs to ensure they can reach their intended networks

Remember to adjust IP addresses, interfaces, and VLAN IDs as needed for your specific network setup.

PBS: Backup Strategy

3-2-1 Backup Setup

In this example, we are backing up a ZFS pool RAID 1 consisting of two 2TB Samsung 990 EVO PRO SSDs.

The goal is to implement the 3-2-1 backup strategy: three copies of your data on two different media, with one copy offsite. This setup is designed for a homelab using consumer software, which adds some challenges due to the lack of enterprise-level scalability. Here's how to do it.

Step 1: Install Proxmox Backup Server (PBS) with Drive Passthrough

First, install PBS with my drive passthrough script. This passthrough drive will be our first backup. Note that while this setup is fine for a homelab, there is a risk in having the backup drive on the host machine that you should be aware of.

Installation Steps:

1. **Access Administration:**

- Go to the browser and navigate to `Administration > Repositories`.
- Add or remove appropriate free and enterprise repositories as necessary.

2. **Update and Reboot:**

- Open the shell and run:

```
apt update
apt upgrade
```

- Reboot the system if a kernel update is applied.

3. **Prepare the Backup Drive:**

- Go to `Administration > Storage > Disks` and wipe the disk you intend to use.

- Then, either create a directory or a ZFS filesystem on this drive.

4. **Add PBS to Proxmox:**

- Go back to the dashboard and copy the fingerprint of the PBS instance.
- Ensure the "Add as datastore" option is checked and note the datastore name you select.

5. **Configure Proxmox:**

- In Proxmox, go to `Datacenter > Storage > Add > Proxmox Backup Server`.
- Set the following values:
  - **ID:** Choose a unique identifier.
  - **Server:** IP address of the PBS instance.
  - **Username:** `root@pam` or the appropriate user.
  - **Password:** Password set in PBS.
  - **Datastore:** Name you noted earlier.
  - **Fingerprint:** Paste the fingerprint you copied.

Now, you can run your backup.

Step 2: Set Up Samba File Share

Since we need to spin up a Windows VM (because Rclone doesn't work well with Proton Drive and Proton Drive only supports NTFS), we'll set up a Samba file share.

Installation and Configuration:

1. **Install Samba:**

```
apt install samba
```

2. **Configure Samba:**

- Edit the Samba configuration file:

```
nano /etc/samba/smb.conf
```

- Add a share definition at the end of the file, assuming you want to share the `/mnt/new-storage` directory (corresponding to the passed-through drive):

```
[SharedDrive]
path = /mnt/new-storage
browseable = yes
```

```
read only = no
guest ok = yes
force user = root
```

- **path:** Directory you want to share.
- **browseable:** Allows the share to be visible when browsing network shares.
- **read only:** Set to `no` to allow writing to the share.
- **guest ok:** Set to `yes` to allow access without a password (optional).
- **force user:** Ensures files are accessible to all users via Samba.

3. **Set Permissions:**

- Ensure that the directory you're sharing has the correct permissions:

```
chmod -R 0777 /mnt/new-storage
```

This makes the directory readable and writable by all users. Adjust permissions as necessary depending on your security requirements.

4. **Restart Samba Service:**

- Apply the changes by restarting the Samba service:

```
systemctl restart smbd
```

Step 3: Access the Share from Windows

1. **Access the Share:**

- Open File Explorer on your Windows PC.
- In the address bar, type `\Proxmox-IP-Address\SharedDrive` and press Enter.
  - Replace `Proxmox-IP-Address` with the IP address of your Proxmox server, and `SharedDrive` with the name of your share as defined in the `smb.conf` file.
  - Authenticate if prompted. If guest access is allowed, it might not ask for credentials; otherwise, use a valid username and password from the Proxmox server.

2. **Map the Network Drive Permanently (Optional):**

- Right-click on "This PC" in File Explorer and select "Map network drive...".
- Choose a drive letter and enter the path to your Proxmox share (e.g., `\Proxmox-IP-Address\SharedDrive`).

- Check "Reconnect at sign-in" and click "Finish".

Step 4: Secure Samba Share with a Password (Optional)

If you want to secure your Samba share with a password, follow these steps:

1. **Create a Samba User:** Add a Linux user if it doesn't already exist:

```
adduser yourusername
```

Add the user to Samba:

```
smbpasswd -a yourusername
```

2. **Modify Samba Configuration:** Edit the Samba configuration file:

```
nano /etc/samba/smb.conf
```

Modify the share definition to require authentication:

```
[SharedDrive]
path = /mnt/new-storage
browseable = yes
read only = no
valid users = yourusername
force user = root
```

3. **Restart Samba Service:** Apply the changes by restarting the Samba service:

```
systemctl restart smbd
```

4. **Access the Share from Windows:** When prompted, enter the username (`yourusername`) and the password set with `smbpasswd`.

By following these steps, you can share the passed-through drive on your Proxmox server with a Windows PC, allowing it to be accessed and used from both systems.