Proxmox VE

Info

Proxmox Scripts | Proxmox Homepage | Proxmox VE Downloads

Proxmox VE is a complete, open-source server management platform for enterprise virtualization. It tightly integrates the KVM hypervisor and Linux Containers (LXC), software-defined storage and networking functionality, on a single platform. With the integrated web-based user interface you can manage VMs and containers, high availability for clusters, or the integrated disaster recovery tools with ease.

Setup

GPU fixes

Machines with GPUs may need tweaking to get working properly.

Installer stuck black screen with GPU

1. When the ISO loads the menu option 'Install Proxmox VE' should be selected press 'e' to start editing the boot option.
2. On the 4th line (starts with linux) add `nomodeset` instead of `quiet`
3. Press Ctrl + x
4. After (attempting to) get DHCP the system will state `Starting a root shell on tty3`
5. Wait for the system to detect the installation failure and drop to shell in TTY1 (Do not switch to TTY3)
6. Run `chmod 1777 /tmp to unlock /tmp`
7. Run `apt update` (to verify that /tmp is unlocked)
8. Run `Xorg -configure` To generate a new configuration file
9. Run `mv /xorg.conf.new /etc/X11/xorg.conf` to move the file to the config directory

10. Edit `/etc/X11/xorg.conf`, search for `nouveau` (there should be only one occurrence) and replace it with `fbdev`
11. Run `startx`
12. Enjoy a working GUI installer
13. Use Ctrl + D to restart the system after the installer exits.

PS: After reboot the screen may keep flickering but remote access should now work and it can probably be fixed by installing the proper drivers.

Networking will fail to initialize with GPU

GPUs are loaded before network, however GPUs may fail to load using nouveau driver. If this is the case then after install the host needs to be tweaked manually over IPMI:

1. `vi /etc/modprobe.d/blacklist.conf`

Text Only

```
blacklist nouveau
blacklist nvidiafb
```

1. `apt-get remove --purge nvidia*`
2. `update-initramfs -u`
3. `reboot`

Proxmox VE No-Subscription Repository

Without an enterprise PVE license, the default apt repo will error. Remove the enterprise apt list `/etc/apt/sources.list.d/pve-enterprise.list` and add the `pve-no-subscription` repo instead. This is done in Ansible management proxmox role. Adding `pve-no-subscription` repo is done using `soft_apt` group variables.

We recommend to configure this repository in /etc/apt/sources.list.

File /etc/apt/sources.list

Text Only

```

```

```
deb http://ftp.debian.org/debian bookworm main contrib
deb http://ftp.debian.org/debian bookworm-updates main contrib

# Proxmox VE pve-no-subscription repository provided by proxmox.com,
# NOT recommended for production use
deb http://download.proxmox.com/debian/pve bookworm pve-no-subscription

# security updates
deb http://security.debian.org/debian-security bookworm-security main
contrib
```

Networking

Using Linux networking

To set up VLANs and required networks from the table above follow the steps from Proxmox Networking docs:

1. Copy `/etc/network/interfaces` to `/etc/network/interfaces.new`: `cp /etc/network/interfaces /etc/network/interfaces.new`
2. Edit `/etc/network/interfaces.new` with changes.
3. Now you have three choices:
4. `cp /etc/network/interfaces.new /etc/network/interfaces` and run `ifreload -a`
5. Go to `<PVE HOST> > System > Network` in the UI and press `Apply Configuration`. This will move changes from the staging interfaces.new file to /etc/network/interfaces and apply them live.
6. (or reboot PVE node, but this is not recommended if not needed)

NOTE: **NEVER reboot networking services manually**, use the tools/methods mentioned in Proxmox Networking docs! Manual restarts WILL break VM networking!

Using netplan

Using netplan is possible, but not recommended as every tap interface attached to a VM has to be added here in order to be persistent. They will not be added dynamically.

The corresponding configuration in /etc/netplan/00-main-conf.yaml might look like this:

Text Only

```
1
2
3
```

```
network:
  version: 2
  ethernets:
    tap113i0: {}
    tap114i0: {}
    tap115i0: {}
    enp129s0f1: {}
    enp5s0f0: {}
  vlans:
    vlan.3701:
      id: 3701
      link: enp129s0f1
  bridges:
    vmbr0:
      addresses:
[172.21.5.87/24]
      gateway4:
172.21.5.254
      interfaces:
[enp5s0f0]
      parameters:
        forward-delay: 0
        stp: false
    vmbr1:
      interfaces:
        - tap114i0
        - tap113i0
        - tap115i0
        - enp129s0f1
      parameters:
        forward-delay: 0
        stp: false
```

User creation

For management purposes, users should be created.

To use Packer with a `packer` user instead of using the root admin privileges for Packer builds (according to packer-plugin-proxmox issue):

- Add user: `pveum useradd packer@pve`
- Set user password: `pveum passwd packer@pve`
- Add role with set privileges: `pveum roleadd Packer -privs "VM.Allocate VM.Clone VM.Config.CDROM VM.Config.CPU VM.Config.Cloudinit VM.Config.Disk VM.Config.HWType VM.Config.Memory VM.Config.Network VM.Config.Options VM.Monitor VM.Audit VM.PowerMgmt Datastore.AllocateSpace Datastore.Allocate`

`Datastore.AllocateSpace Datastore.AllocateTemplate Datastore.Audit Sys.Audit VM.Console SDN.Allocate SDN.Audit SDN.Use"`

- GUI: for role privs: navigate to the Permissions → Roles tab from Datacenter and click on the Create button. There you can set a role name and select any desired privileges from the Privileges drop-down menu.
- Add role to user: `pveum aclmod / -user packer@pve -role Packer`

To use Terraform with `terraform` user (according to Telmate provider doc):

- Add user: `pveum useradd terraform@pve`
- Set user password: `pveum passwd terraform@pve`
- Add role with set privileges: `pveum roleadd Terraform -privs "Datastore.AllocateSpace Datastore.Audit Pool.Allocate Sys.Audit Sys.Console Sys.Modify VM.Allocate VM.Audit VM.Clone VM.Config.CDROM VM.Config.Cloudinit VM.Config.CPU VM.Config.Disk VM.Config.HWType VM.Config.Memory VM.Config.Network VM.Config.Options VM.Migrate VM.Monitor VM.PowerMgmt SDN.Allocate SDN.Audit SDN.Use"`
- GUI: for role privs: navigate to the Permissions → Roles tab from Datacenter and click on the Create button. There you can set a role name and select any desired privileges from the Privileges drop-down menu.
- Add role to user: `pveum aclmod / -user terraform@pve -role Terraform`

API token can be generated in the UI from `Datacenter` tab from `Permissions > API Tokens` menu and then from the button `Add`.

User Management

Disk pools

Root storage is created on installation using with ZFS and RAID by default.

LVM-thin storage for VMs was created on 15K SAS disks:

1. Wipe disks that you wish to use: `wipefs -a /dev/sda /dev/sdb /dev/sdc /dev/sdd`
2. Create physical volumes: `pvcreate /dev/sda /dev/sdb /dev/sdc /dev/sdd`
3. Create volume group: `vgcreate vgrp /dev/sda /dev/sdb /dev/sdc /dev/sdd`
4. Create thin-pool on the volume group: `lvcreate -L 2T --thinpool thpl vgrp`
5. From `Datacenter -> Storage`, add an `LVM-Thin` storage

Directory storage for backups was created on 4TB HDDs:

1. Wipe disks that you wish to use: `wipefs -a /dev/sde /dev/sdf`
2. Create mirrored pool with disks: `zpool create tank mirror /dev/sde /dev/sdf`
3. Disk mount points are bound to change after reboots or hardware changes, resulting in ZPOOL degradation. Hence we will make the zpool use block device identifiers (/dev/disk/by-id) instead of mount-points.
4. Export pool: `zpool export tank`
5. `zpool import -d /dev/disk/by-id tank`
6. Importing back with /dev/disk/by-id immediately will seal the disk references.
7. Add it to PVE for storage: `pvesm add zfspool tank -pool tank`
8. Create mountpoint on the zpool: `zfs create tank/bkup -o mountpoint=/bkup`
9. From the GUI via `Datacenter -> Storage -> Add -> Directory` add a Directory storage with the above `mountpoint` as the mount.

Mirrored ZFS pool for VMs was created with new 4TB SSDs from the UI `Datacenter -> <node> -> Storage -> ZFS -> Create` with following parameters: - Name: ssdpool - RAID: RAID10 - Compression: lz4 (recommended by Proxmox documentation) - ashift: 12 (Ashift tells ZFS what the underlying physical block size your disks use is. It's in bits, so ashift=9 means 512B sectors (used by all ancient drives), ashift=12 means 4K sectors (used by most modern hard drives), and ashift=13 means 8K sectors (used by some modern SSDs). CT4000MX500 has sector size as follows - logical/physical: 512 bytes / 4096 bytes.) - Devices: `/dev/sdi /dev/sdj /dev/sdk /dev/sdl`

Clustering

Proxmox supports clustering. For that to be useful, the modes have to have mostly mirroring setups. This applies especially for networking and storage. Migration between nodes is not possible if the network hardware is set to bridges that do not exist on the other machine.

Administration

"Host key verification failed. Failed to run vncproxy."

Make sure `/etc/ssh/ssh_known_hosts` is up to date. Clean the file and add up to date values for both IPs and hostnames of each PVE node on all PVE nodes manually.

VM creation

Main method for creating VMs in Proxmox is via cloning existing VM templates that were built using Packer.

To create a clone of a VM template manually from UI:

- Use a FQDN for the VM name, e.g. `vm-ubuntu-20-04-1.hpc.taltech.ee` as this will be used as the hostname for the VM, but `vm-ubuntu-20-04-1` would not be used as the hostname, instead the template's hostname would be used.
- Create as a Full Clone not a Linked Clone
- Change the VM ID to a unique ID, default PVE suggestion should already be fine.
- Use Cloud Init for the VM configuration; to change IP, DNS, gateway, SSH keys.

To create a clone of a VM template manually from the CLI:

- Read Proxmox Cloud-Init Support docs and Cloud-Init FAQ

ISO Upload

ISO images can be uploaded to Proxmox in two ways:

1. Via CMD and placed in `/var/lib/vz/template/iso`
2. `cd /var/lib/vz/template/iso && wget <iso_url>`
3. Via the Proxmox UI from `local (Storage Pool) -> ISO Images -> Upload / Download from URL`.

VM Template Builds (and ISO Upload) with Packer

VM templates from ISO images for Proxmox VMs are built using the jamlab-packer repository using Hashicorp Packer.

VM Provisioning with Terraform

VMs are provisioned using Packer VM templates from the jamlab-terraform repository using Hashicorp Terraform.

VM procedures

Disk increase

On the PVE host follow the official documentation procedure to increase the disk size of the VM you wish.

Then inside the VM:

Bash

```
# Check which partition to resize
sudo fdisk -l
# Grow partition (notice the space!)
sudo growpart /dev/sda X
# Resize to fill partition (notice the missing space this time!)
sudo resize2fs /dev/sdaX
```

VM Disk Migration

VM live storage move migration (no downtime)

First, just in case, a backup snapshot was taken: `<VM> -> Backup -> Backup Now`

Then the VM migration was started from `<VM> -> Hardware -> Select Disk (verify the right one is highlighted and active) -> Disk Action -> Move Storage` (VM was not shut down, migration was done live, while the VM was running)

Note: cloud-init drive can not be moved similarly, for that the backup restore method must be used.

VM backup restore migration (with downtime)

A backup must exist, if it does not, then follow: `<VM> -> Backup -> Backup Now`

To restore, follow: `<VM> -> Backup -> Restore`

Then select storage to restore to, check `Unique` if you wish to destroy the current one (if it exists), check `Start after restore` if you wish to immediately start it.

Note: the restore process creates a new cloud-init drive that is also on the same storage as the restore storage that was selected.

Downtime for the test VM with a fairly empty and small disk was less than 30 seconds, results may vary depending on the disk size.

VM Migration To Another Node

Both servers have to be exact same versions. This includes packages and kernel.

Upgrade, reboot if necessary, check versions on all nodes with: `pveversion -v`

Note: if interfaces differ on nodes, you must temporarily replace them with something that exists on both nodes.

Foreign QEMU/KVM migration to Proxmox

To migrate a foreign VM disk to Proxmox:

1. Copy the foreign VM disk image somewhere on the Proxmox host you wish to migrate to
2. Use the following example to migrate:

Bash

```
VM_ID=666 #CHANGE ME!
VM_IMAGE=/root/migr-disks/myimage.qcow2 #CHANGE ME!

qm create ${VM_ID}
qm disk import ${VM_ID} ${VM_IMAGE} local-zfs
qm set ${VM_ID} --scsi0 local-zfs:vm-${VM_ID}-disk-0
qm set ${VM_ID} --boot order=scsi0
```

VM with Nvidia GPU passthrough

Some resources on this:

- The Ultimate Beginner's Guide to GPU Passthrough (Proxmox, Windows 10) and its shorter version Proxmox PCI(e) Passthrough in 2 minutes
- PCI Passthrough
- How to Install Nvidia Drivers on Rocky Linux 9 or 8

Creating the VM

Create a new VM with following settings.

SCSI Controller: `VirtIO SCSI Single`

BIOS: `OMVF` (UEFI, for PCIE support)

Machine: `q35` (for PCIE support)

Processor type: `host` or according to your architecture if that does not work

Make sure VM has at least 20GB of space in boot disk.

Configuring the VM

After VM is created change some settings.

Under `Hardware` press `Add` and add a new PCI device, from the `Raw device` option select your GPU to pass through.

Set `Display` to `VirtIO-GPU`

Set `PCI Device` as `Primary GPU`

Then boot the VM and from UEFI settings disable secure boot options and boot.

Inside the VM set up the drivers as follows:

For Ubuntu: `sudo apt install nvidia-driver-535 nvidia-dkms-535`

For RedHat:

1. Activate the CodeReady Builder (CRB): `sudo dnf config-manager --set-enabled crb`
2. Install EPEL: `sudo dnf install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm https://dl.fedoraproject.org/pub/epel/epel-next-release-latest-9.noarch.rpm`
3. Add Nvidia Repository: `sudo dnf config-manager --add-repo http://developer.download.nvidia.com/compute/cuda/repos/rhel9/$(uname -i)/cuda-rhel9.repo`
4. Add dependencies: `sudo dnf install kernel-headers-$(uname -r) kernel-devel-$(uname -r) tar bzip2 make automake gcc gcc-c++ pciutils elfutils-libelf-devel libglvnd-opengl libglvnd-glx libglvnd-devel acpid pkgconfig dkms`

5. Install Nvidia driver module and cuda:

Bash

```
dnf module install nvidia-driver:535-dkms
dnf install cuda-12-2
```