Proxmox

- Proxmox VM Template & Cloud-Init auto install
- Proxmox Ceph Installation & Konfiguration
- Nvidia VGPU Driver installation

Proxmox VM Template & Cloud-Init auto install

Voraussetzungen

- **Proxmox VE** ≥ 6.x
- Linux-Cloud-Image (Ubuntu, Debian, Alma, Rocky, …)
- SSH-Key auf deinem Client

1. Cloud-Image herunterladen

```
wget <https://cloud-images.ubuntu.com/noble/current/noble-server-cloudimg-amd64.img>
```

□□ Wichtig: **Normale ISOs funktionieren nicht**, nur *Cloud Images*.

Weitere Ubuntu Cloud Images gibts hier: https://cloud-images.ubuntu.com/

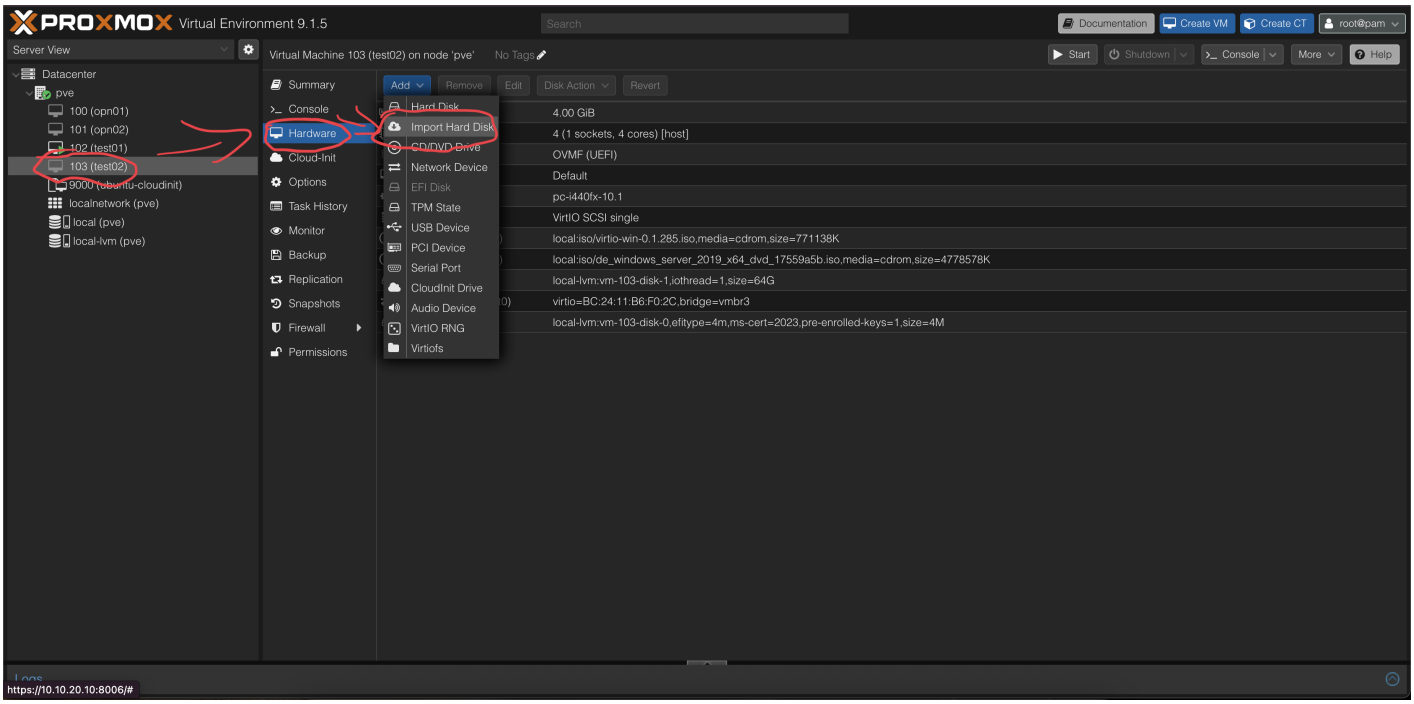
2. VM als Cloud-Init-Template anlegen

```
qm create 9000 \\
  --name ubuntu-cloudinit \\
  --memory 2048 \\
  --cores 2 \\
  --net0 virtio,bridge=vmbr0
```

Disk importieren:

```
qm importdisk 9000 noble-server-cloudimg-amd64.img local-lvm
```

Wichtige Information: Bei Proxmox version 9.1.4 und neuer gibt es einen bug wenn man den VM Speicher als LVM-Thin formatiert hat. In dem Fall muss die Image Datei unter local (pve) → Import → Upload hochgeladen werden.



Im Anschluss muss das Hochgeladene image nur noch Hard Disk importiert werden.

Disk & Boot setzen:

```
qm set 9000 \\
  --scsihw virtio-scsi-pci \\
  --scsi0 local-lvm:vm-9000-disk-0 \\
  --boot c \\
  --bootdisk scsi0
```

Cloud-Init aktivieren:

```
qm set 9000 --ide2 local-lvm:cloudinit
qm set 9000 --serial0 socket --vga serial0
```

Cloud-Init konfigurieren (Proxmox GUI)

Dies ist nur erforderlich wenn man nicht mit Snippets arbeitet

Wenn man aber mit Snippets arbeitet ist es ratsam im Template in IP-Config die Konfiguration auf DHCP zu stellen.

Wenn trotzdem eine statische IP für einen Dienst gewünscht wird, kann man dies auch über das Snippet konfigurieren.

In der VM unter **Cloud-Init**:

- □□ User: `ubuntu`
- □□ SSH Public Key einfügen
- □□ IP-Config:
  - DHCP **oder**
  - `ip=192.168.1.50/24,gw=192.168.1.1`
- □□ DNS optional

VM zum Template machen

```
qm template 9000
```

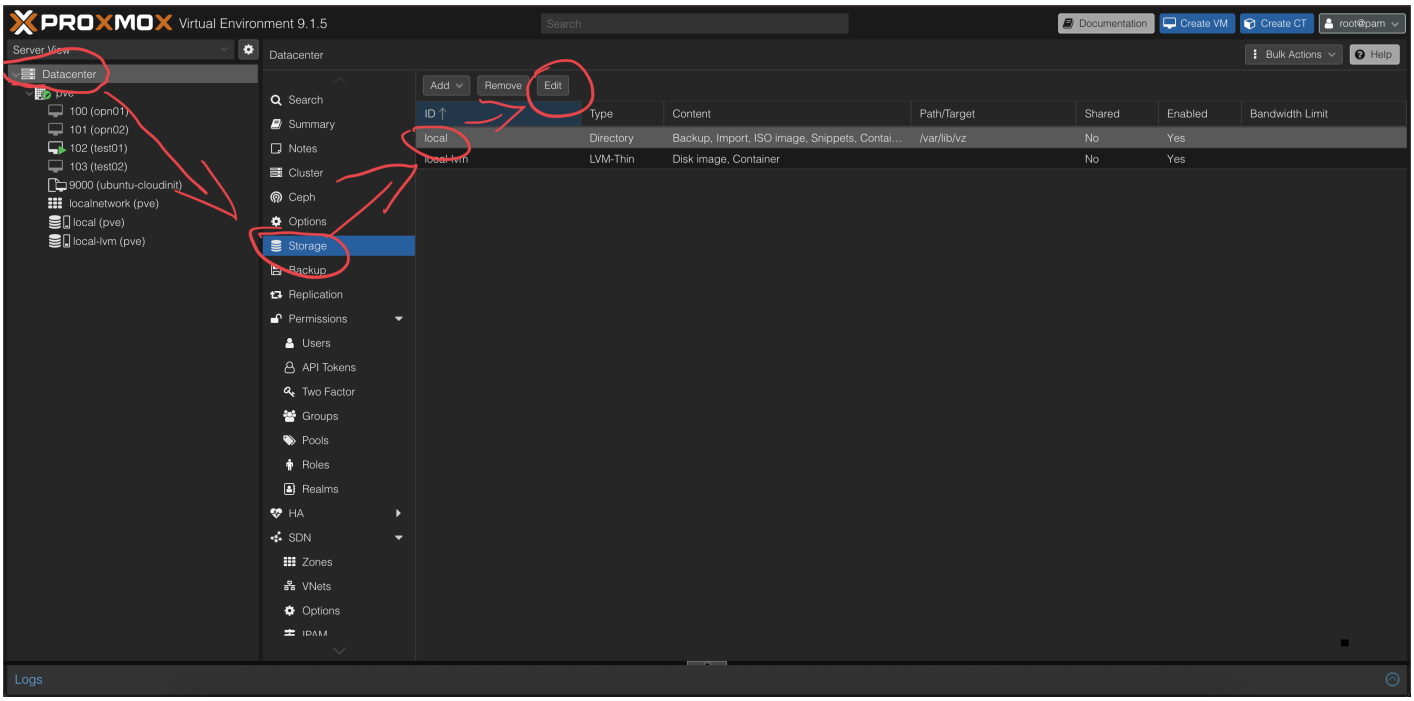
□□ Fertig! Das Golden Image ist bereit.

Snippets erstellen und einbinden

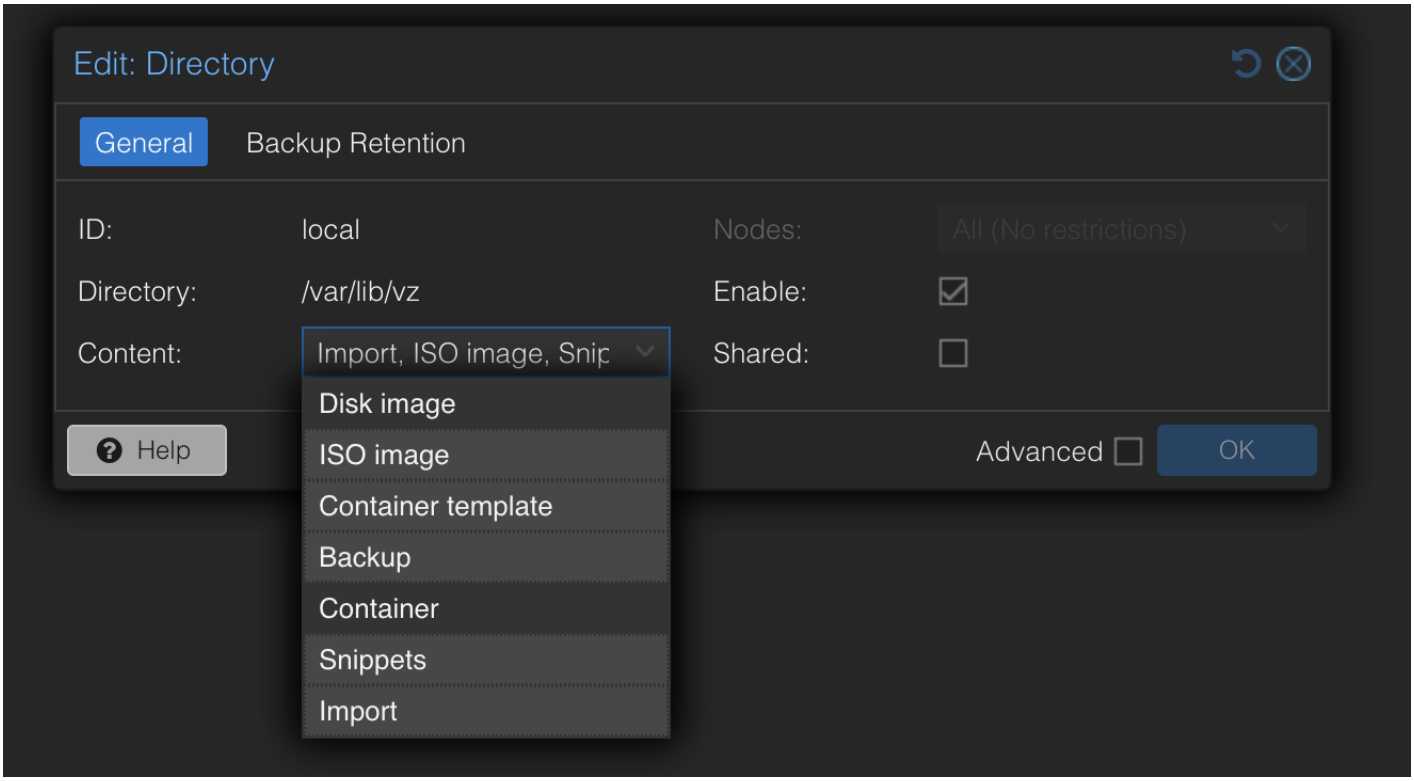
Snippet Speicherort einstellen

Bevor wir Snippets verwenden können, muss erst einmal das Speichern von Snippets erlaubt werden.

Dafür im Proxmox Webinterface unter dem Punkt Datacenter → Storage → local → Edit



Hier muss nur im Content Menü Snippets angeklickt werden und dann nur noch mit OK bestätigt werden.



Nun muss über die Proxmox shell nur noch die Datei am Richtigen Ort abgelegt werden

Snippets liegen dann hier:

```
/var/lib/vz/snippets/
```

user-data Datei erstellen

Beispiel:

```
nano /var/lib/vz/snippets/user-data-web.yaml
```

Inhalt:

```
#cloud-config

hostname: web01

users:
  - name: devops
    shell: /bin/bash
    sudo: ['ALL=(ALL) NOPASSWD:ALL']
    ssh_authorized_keys:
      - ssh-ed25519 AAAA...DEINKEY

package_update: true
packages:
  - nginx
  - curl

runcmd:
  - systemctl enable nginx
  - systemctl start nginx
  - echo "Provisioned by cloud-init" > /etc/motd
```

Wichtig:

- Erste Zeile **muss** `#cloud-config` sein
- YAML korrekt eingerückt!

VM klonen (vom Template)

```
qm clone 9000 101 --name web01
```

Custom user-data anhängen

```
qm set 101 --cicustom "user=local:snippets/user-data-web.yaml"
```

VM Starten

```
qm start 101
```

Fertig □□

Beim ersten Boot wird die Config ausgeführt.

Erweiterte Beispiele

Weitere Beispiele habe ich auf meiner Cloud liegen.

Proxmox

Proxmox Ceph Installation & Konfiguration

1. Vorbereitung & Anforderungen

Bevor du beginnst, solltest du sicherstellen, dass folgende Voraussetzungen erfüllt sind:

? Hardware-Empfehlungen

- Mindestens **3 physische Proxmox-Nodes** für Quorum und Redundanz.
- Leistungsfähige **CPU & genügend RAM** (Ceph benötigt insbesondere RAM pro OSD).
- Netzwerk mit **dedizierten Ceph-Verbindungen** (z. B. 10 Gbps oder mehr).
- Direkter Zugriff auf Festplatten (kein RAID) – Ceph arbeitet besser mit HBAs.

□□ Grundidee von *Hyper-Converged Infrastructure (HCI)*: Compute **und** Storage laufen gemeinsam auf denselben Servern.

---

2. Ceph Installation

Du kannst Ceph entweder über die Proxmox-Weboberfläche oder per CLI installieren.

? Installation über Web Wizard (empfohlen)

1. Öffne die Proxmox Web-GUI.
2. Wähle einen Cluster-Node aus.
3. Navigiere zu **Ceph → Installieren**.
4. Folge dem Assistenten:
   ➤ Wähle die Ceph-Version
   ➤ Bestätige Installation
   ➤ Setze Netzwerkeinstellungen (Public & optional Cluster Network)
5. Nach Abschluss wird Ceph installiert und betriebsbereit sein.

Netzwerkeinstellungen:

- **Public Network:** Ceph Datenverkehr (z. B. Client/Replication)
- **Cluster Network (optional):** Intern für OSD Replikation (empfohlen separat).

? Installation über CLI

Wenn du lieber die Konsole nutzt:

```
pveceph install
```

Danach die Erstkonfiguration:

```
pveceph init--network <CEPH_PUBLIC_NETWORK>/<CIDR>
```

Dadurch wird `/etc/pve/ceph.conf` erzeugt und automatisch auf alle Cluster-Nodes verteilt.

---

3. Ceph Dienste erstellen

Nach der Installation musst du die wichtigen Ceph-Dienste einrichten:

Ceph Monitors (MON)

Mindestens **3 Monitore** für Quorum und Ausfallsicherheit:

```
pveceph mon create
```

oder über GUI: **Ceph → Monitor → Create**.

Ceph Manager (MGR)

Ein Manager ist wichtig für Cluster-Überwachung:

```
pveceph mgr create
```

oder über GUI: **Ceph → Manager → Create**.

---

4. OSDs erstellen

Ceph speichert Daten über OSD-Dienste. Empfohlen: **1 OSD pro physischer Festplatte**.

? OSD per CLI

```
pveceph osd create /dev/sdX
```

→ Ersetze `/dev/sdX` durch das entsprechende Block-Device.

◫ Falls eine Festplatte vorher schon Ceph-Daten enthält:

```
ceph-volume lvm zap /dev/sdX--destroy
```

⚠ Alle Daten auf dem Laufwerk werden gelöscht.

---

5. Pool erstellen

Ein *Pool* ist eine logische Einheit für Ceph-Speicher.

```
pveceph pool create <pool-name>--add_storages
```

- Größe (`size`): Replicas (Standard: 3)
- Anzahl der Placement Groups (`pg_num`): Bestimmt Leistung und Verteilung

Pools erscheinen dann automatisch in der Proxmox-GUI als Speicherziel.

---

6. Ceph in Proxmox hinzufügen

Nachdem Pools erstellt sind:

1. In der Proxmox-GUI gehe zu **Datacenter → Storage → Add → RBD**.
2. Wähle den Ceph-Pool aus.
3. Gib die Key-Ring und Mon Host-Informationen an.

→ Proxmox kann dann VM-Disks direkt auf Ceph speichern (RBD).

---

7. Optional: CephFS konfigurieren

Wenn du ein verteiltes Dateisystem möchtest:

```
pveceph mds create
pveceph fs create--pg_num128--add-storage
```

CephFS benötigt mindestens einen MDS.

---

8. Wartung & Monitoring

Du kannst den Ceph-Status überwachen:

```
ceph-s
watch ceph--status
```

Fehler können z. B. durch Netzwerkprobleme, ausgefallene OSDs oder unzureichende Ressourcen entstehen.

Tipps aus den Empfehlungen

→ Vermeide RAID-Controller und nutze direkte HBAs.

→ Reserve genug RAM pro OSD (z. B. 1 GiB pro TiB).

→ Dedizierte Netzwerkschnittstellen für Ceph-Traffic bringen bessere Performance.

Nvidia VGPU Driver installation

0. Vorbereitung – Proxmox & GPU-Check

Bevor irgendetwas installiert wird, sollte geprüft werden, ob die verwendete NVIDIA-GPU **vGPU-fähig** ist. Dazu gibt man den Chipsatznamen (z. B. „1060“ oder „2080“) ein. Unterstützte Karten erkennt das Skript entsprechend.

Wenn mehrere GPUs eingebaut sind, muss eine davon für das **Passthrough** reserviert werden und die andere für vGPU genutzt werden.

---

1. BIOS & Skript starten

1. **VT-d / IOMMU im BIOS aktivieren**
   – Intel: *VT-d*
   – AMD: *IOMMU*
2. Server neu starten und per SSH anmelden.
3. Skript herunterladen und ausführen:
   git clone https://github.com/wvthoog/proxmox-vgpu-installer.git
   cd proxmox-vgpu-installer
   bash proxmox-installer.sh

   13zRMxL.gif
4. Im Menü wählt man z. B. „New vGPU installation“, um neu zu installieren.
   Nach Abschluss startet man das System neu.

---

2. Installation fortsetzen

Nach dem Neustart erneut das Skript starten.
Jetzt wird geprüft, ob **VT-d / IOMMU geladen ist** und eine Nvidia-Karte gefunden wurde.
Anschließend wählt man die gewünschte **Treiber-Version** für Proxmox 7.x oder 8.x.

bmV4AN9.gif

Das Skript lädt den vGPU-Host-Treiber, patched ihn und installiert ihn.
Am Ende erhält man zwei URLs: eine für den Linux-Gasttreiber und eine für den Windows-Gasttreiber. Diese werden später in den VMs benötigt.

---

4. vGPU einer VM zuweisen

Nach der Installation erzeugt ein Befehl (`mdevctl types`) verschiedene vGPU-Profile. Diese Profile teilen den VRAM entsprechend auf (z. B. 4 GB, 2 GB, 1 GB).

So weist man eine vGPU in der Proxmox-GUI zu:

1. VM auswählen
2. Reiter „Hardware“ öffnen
3. „Add“ → „PCI Device“ klicken
4. Nvidia-GPU auswählen (als „Mediated Device“)
5. Gewünschtes vGPU-Profil wählen
6. Hinzufügen und speichern

---

5. Gast-Treiber installieren

Linux-Gast

1. System aktualisieren:
   sudo apt update && sudo apt dist-upgrade

2. Kernel-Headers installieren:
   sudo apt install linux-headers-$(uname -r)

3. Den vom Skript bereitgestellten Nvidia-Grid-Treiber herunterladen und installieren:
   chmod +x NVIDIA-…-grid.run
   sudo ./NVIDIA-…-grid.run --dkms

4. Mit `nvidia-smi` prüfen, ob die vGPU läuft.HJON9hr.png

Windows-Gast

cjERasO.gif

Vor der vGPU-Treiberinstallation sollte ein vorhandener Nvidia-Treiber vollständig entfernt werden (z. B. über DDU). Danach den passenden GRID-Treiber installieren.

---

6. Tipps & Zusatzfunktionen

Das Skript unterstützt zusätzliche Argumente wie `–debug`, `–step`, `–url` oder `–file` zur flexibleren Nutzung. Außerdem gibt es eine Liste möglicher Verbesserungen und eine **Changelog-Übersicht** mit den wichtigsten Updates seit der Veröffentlichung.

---

7. Fehlerbehebung

Tritt ein Problem auf, kann man meist den vGPU-Installer entfernen, den Server neu starten und den Vorgang erneut beginnen. Falls das nicht hilft, lohnt sich ein Blick in die `debug.log` oder eine Rückmeldung an den Autor zur Verbesserung des Skripts.