

yoTech24

Proxmox VE

Lernbereich-Zusammenfassung · 15 Module (M0–M14)

Für Admins und IT-Dienstleister, die von VMware vSphere oder Microsoft Hyper-V zu Proxmox VE migrieren. Referenz-Szenario: 1–2 Nodes, 4–6 VMs, 1–3 LXC-Container.

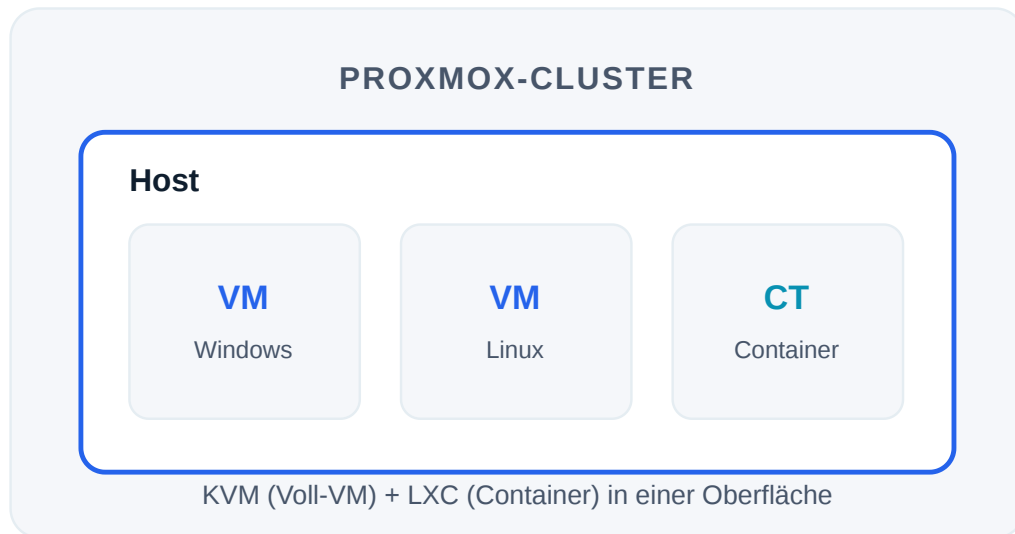
Diese PDF fasst die Kernpunkte jedes Moduls zusammen. Die vollständigen interaktiven Folien inkl.

Audio-Dialogen (Podcast) finden sich unter

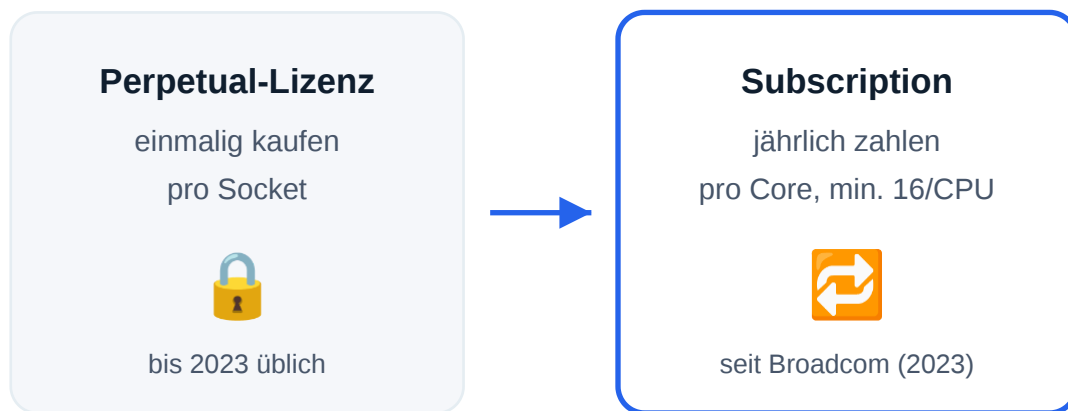
yotech24.com/akademie/themen/spezialisierung-proxmox-ve/

M0 · Intro, Marktlage & Ausblick

Für Admins, die von VMware/Hyper-V migrieren – wo steht der Markt 2026, wo steht Proxmox, und welche Alternativen gibt es wirklich?



Marktlage: VMware unter Broadcom



Für viele KMU-Admins war das kein gewöhnliches Software-Update, sondern ein handfestes Budgetproblem: Aus einer einmaligen Investition wurde eine wiederkehrende Abo-Position – bei spürbar höherem Betrag.

- Übernahme von VMware durch Broadcom 2023 – ein Halbleiter- und Software-Konzern, der zugekaufte Produktlinien erfahrungsgemäß auf wenige, margenstarke Kernprodukte konsolidiert; VMware wurde in dieses Muster eingeordnet
- Perpetual-Lizenzen (einmal gekaufte Nutzungsrechte plus optionales Support-Abo) abgeschafft → nur noch Subscription; bereits gekaufte **Perpetual**-Lizenzen laufen zwar technisch weiter, erhalten aber keine neuen Sicherheitsupdates mehr, sobald der zugehörige Support-Vertrag endet
- Abrechnung jetzt per Core statt pro Socket, mit einem Minimum von 16 Kernen pro CPU – ein Server mit zwei 8-Kern-Prozessoren wird lizenztechnisch so behandelt, als hätte er 32 Kerne; hinzu kommt Bundle-Zwang: einzelne Produkte wie vSphere sind nicht mehr separat erhältlich, sondern nur noch im Paket **VCF** (VMware Cloud Foundation) oder **VVF** (VMware vSphere Foundation)
- Broadcom lässt bei Bestandskunden mit ausgelaufenen Perpetual-Lizenzen verstärkt Lizenz-Audits durchführen – wer nicht rechtzeitig auf ein gültiges Abo umgestiegen ist, riskiert Nachzahlungen zu den neuen, deutlich höheren Listenpreisen

Alternativen-Landschaft

PLATTFORM	REIFEGRAD	ZIELGRUPPE	PREISMODELL	HA/CLUSTER	ÖKOSYSTEM	MIGRATION
Proxmox VE	hoch	KMU, Homelab	Open-Source + Abo	ja (Corosync)	wachsend, breit	niedrig-mittel
Nutanix AHV	hoch	Enterprise	kommerziell	ja (AOS)	groß, HCI-fokus	mittel-hoch
XCP-ng	mittel	Xen-erfahren	Open-Source + Abo	ja (nativ)	klein, wachsend	mittel
Hyper-V	hoch	Windows-Haus	in Windows Server	ja (WSFC)	groß (Windows)	niedrig (Windows)
SUSE Harvester	wachsend	Kubernetes-nativ	Open-Source + Abo	ja (Longhorn)	klein, Cloud	hoch

Entscheidungshilfe: Proxmox VE passt, wenn Lizenzkosten und Bedienbarkeit für Linux-affine Admins im Vordergrund stehen und der Betrieb überschaubar bleibt (wenige bis mittlere Node-Zahl, kein 24/7-Enterprise-SLA nötig). Nutanix AHV oder eine tiefe VMware-Alternative im Enterprise-Segment lohnt sich, wenn vertraglich fixierter 24/7-Support und eine durchintegrierte HCI-Suite Pflicht sind. Reines Windows-Umfeld ohne Linux-Anteil spricht für Hyper-V, bereits vorhandene Kubernetes-Strategie spricht für SUSE Harvester, und wer aus dem Xen-Umfeld kommt, findet in XCP-ng den direktesten Umstieg.

- **Proxmox VE:** quelloffen mit optionalem Support-Abo, sehr aktive Community, starkes KMU- und Mittelstandsumfeld – Kernstärke ist der niedrige Einstieg samt vollem Funktionsumfang ohne Lizenzkosten, Kerneinschränkung ist das im Vergleich zu VMware jüngere Enterprise-Ökosystem an Drittanbieter-Tools
- **Nutanix AHV:** technisch ausgereifte Hyperconverged-Plattform, klar auf Enterprise-Kunden zugeschnitten – Kernstärke ist die tiefe Integration von Compute, Storage und Management aus einer Hand, Kerneinschränkung ist der hohe Preis und die Bindung an die eigene Hardware-/Software-Suite
- **XCP-ng:** freier, quelloffener Fork des Citrix-/Xen-Hypervisor-Codes mit ähnlichem Lizenzmodell wie Proxmox – Kernstärke ist die Xen-Reife für erfahrene Anwender, Kerneinschränkung ist die kleinere Community und das dünnere Angebot an fertigen Backup-/Management-Integrationen
- **Hyper-V:** fester Bestandteil von Windows Server, damit ohne zusätzliche Lizenzkosten und ohne neue Betriebssystem-Lernkurve verfügbar – Kernstärke ist die nahtlose Einbindung in reine Windows-Umgebungen, Kerneinschränkung ist die schwächere Unterstützung gemischter Linux/Windows-Szenarien
- **SUSE Harvester:** Kubernetes-native HCI-Lösung (aufbauend auf KubeVirt und Longhorn), cloud-nativ statt klassischer Hypervisor-Denke – Kernstärke passt, wenn ohnehin auf Container-Infrastruktur gesetzt wird, Kerneinschränkung ist das andere Betriebsmodell mit entsprechend höherem Umstellungsaufwand für klassische VM-Admins

Ist Proxmox das neue ESXi für KMU?



KMU

wenige Hosts, gemischt

passt sauber



Großenterprise

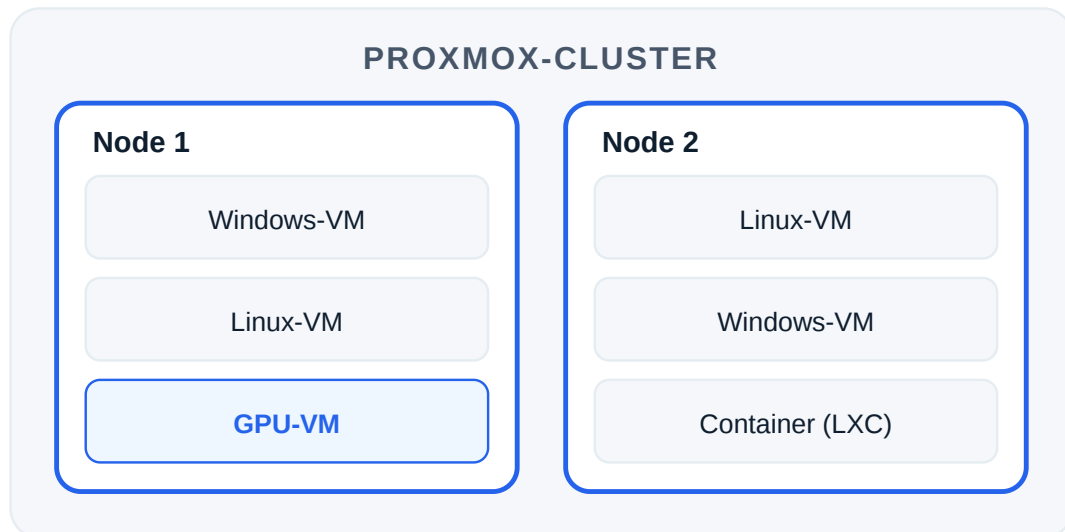
>100 Hosts, 24/7-SLA

passt bedingt

Referenz-Szenario dieses Lernbereichs: ein typischer KMU-Betrieb mit **1–2 Nodes**, **4–6 VMs** (z. B. Fileserver, Active Directory, ein bis zwei Fachanwendungen, interner Webserver) und **1–3 LXC-Containern** für schlanke Zusatzdienste (Reverse Proxy, interner DNS, Monitoring-Agent). Auf dieses Szenario beziehen sich die konkreten Empfehlungen in den folgenden Modulen zu Kosten, Redundanz und Skalierung – wo ein anderer Rahmen gemeint ist, steht das explizit dabei.

- In klassischen KMU-Szenarien – ein bis wenige Hosts, überschaubare VM-Anzahl, keine hochkomplexen Storage-Anforderungen – ersetzt Proxmox VE ESXi funktional nahezu lückenlos: Snapshots, Live-Migration, Backup und Cluster-Betrieb sind alle vorhanden
- Grenzen zeigen sich dort, wo das VMware-Ökosystem historisch stark ist: Drittanbieter-Tools, die tief in vCenter integriert sind (bestimmte Backup-, Monitoring- oder Compliance-Lösungen), müssen ersetzt oder neu angebunden werden; ebenso fehlt ein direktes Pendant zu herstellergarantiertem 24/7-Enterprise-Support mit vertraglich fixierten Reaktionszeiten, wie ihn große Konzerne für unternehmenskritische Systeme fordern
- Der eigentliche Umstieg ist weniger technisch als mental: Proxmox versteckt das darunterliegende Debian-Linux nicht hinter einer geschlossenen Appliance, sondern macht es sichtbar – wer bisher nie eine Kommandozeile auf dem Hypervisor gebraucht hat, sieht sie hier gelegentlich, muss sich dafür aber nicht an ein völlig neues, fremdes Bedienkonzept gewöhnen
- Bei diesem Umfang (1–2 Nodes, 4–6 VMs, wenige Container) bleibt der Betrieb für einen einzelnen internen Admin oder einen Dienstleister im Teilzeit-Support gut überschaubar – der Komplexitätssprung kommt erst mit echtem Multi-Node-Cluster-Betrieb (Modul 7) oder Ceph-Storage (Modul 3), nicht schon mit der Plattformwahl selbst

Gemischt & KI-tauglich



- Ein einzelner Proxmox-Cluster bedient Windows-Server-VMs (z. B. Active Directory, Fileserver, Fachanwendungen) und Linux-VMs oder -Container (Webserver, Datenbanken, interne Tools) gleichberechtigt nebeneinander – kein separates Produkt, keine getrennte Verwaltung nötig
- Für KI-Workloads lässt sich eine GPU per PCI(e)-Passthrough direkt einer VM zuweisen, sodass diese die volle Rechenleistung exklusiv nutzt – etwa für ein lokal gehostetes LLM oder Bildgenerierung im eigenen Haus statt in der Cloud; Grenzen bestehen bei geteilter GPU-Nutzung zwischen mehreren VMs (vGPU), die je nach Kartenmodell und Treiberunterstützung nur eingeschränkt oder gar nicht funktioniert
- Voraussetzung für GPU-Passthrough ist ein IOMMU-fähiges System (siehe Modul 2, „CPU-Voraussetzungen“); wer eine GPU stattdessen per **vGPU** auf mehrere VMs gleichzeitig aufteilen will, braucht dafür herstellerabhängig meist eine Profi-/Datacenter-Karte samt zusätzlicher Lizenz (z. B. NVIDIA vGPU) – reine Consumer-GPUs lassen sich trotzdem ganz regulär per Passthrough exklusiv durchreichen, nur eben nicht unter mehreren VMs teilen
- Damit deckt ein einziges Proxmox-Cluster das typische KMU-Szenario vollständig ab: klassische Windows/Linux-Infrastruktur und experimentelle KI-Workloads auf derselben Plattform, ohne zusätzliche Spezial-Hardware oder ein zweites Virtualisierungsprodukt

Zukunft & nächster Schritt



- Der Marktanteil von Proxmox VE zeigt seit der Broadcom-Übernahme eine klar steigende Tendenz, getragen von einer aktiven Community (Forum, Bugtracker, regelmäßige Major-Releases) und wachsender Verbreitung bei Systemhäusern und Hosting-Anbietern
- Statt unter Zeitdruck direkt produktive VMware-Umgebungen umzuziehen, empfiehlt sich der umgekehrte Weg: zuerst ein isoliertes Testlabor aufbauen (auch als Nested-Virtualization-Setup auf vorhandener Hardware möglich), dort Migrationsschritte und Konfiguration üben, danach eine einzelne unkritische VM als Pilot migrieren – erst wenn das reibungslos läuft, folgt der produktive Rollout
- Dieser Modul-0-Überblick war die Landkarte; ab Modul 1 geht es in die technischen Grundlagen und die Architektur von Proxmox VE im Detail

M1 · Grundlagen & Architektur

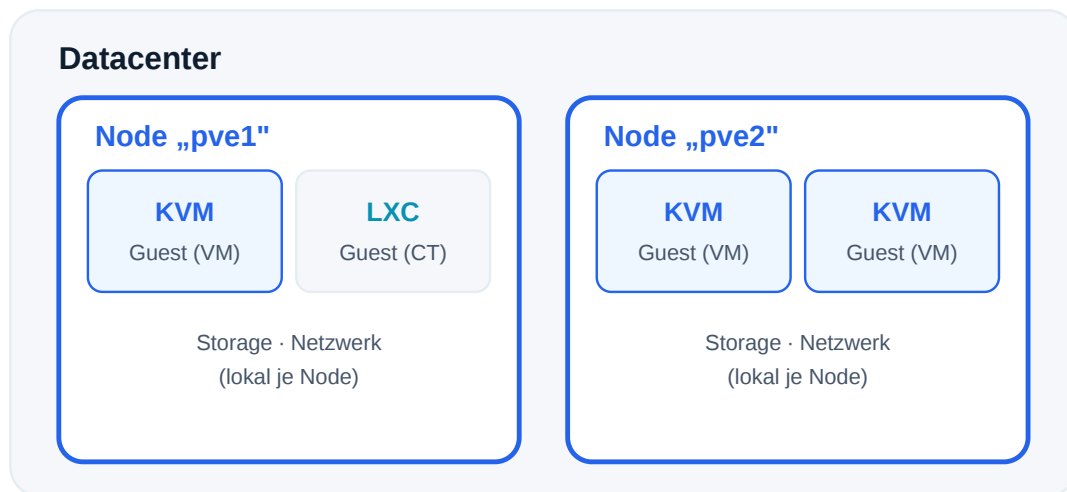
Vom Blech zur virtuellen Maschine – wie Proxmox VE aufgebaut ist

Was ist Proxmox VE?

Proxmox VE ist Debian-Linux, erweitert um eine Weboberfläche für Virtualisierung – kein separates Betriebssystem, sondern ein sehr gut konfiguriertes Debian.

- Basis: aktuelles Debian (Long-Term-Support-Kernel, regelmäßige Sicherheitsupdates aus dem Debian- und Proxmox-Repository) – wer schon einmal einen Debian-Server administriert hat, kennt also bereits die Grundlage
- KVM/QEMU für vollwertige virtuelle Maschinen – jedes Gast-Betriebssystem, volle Hardware-Emulation wo nötig, native Geschwindigkeit dank Hardware-Virtualisierung (Intel VT-x/AMD-V)
- LXC für leichtgewichtige Linux-Container – teilen sich den Host-Kernel, starten in Sekunden statt Minuten, ideal für schlanke Linux-Dienste ohne den Overhead einer vollen VM
- Verwaltung wahlweise über Web-GUI, Kommandozeile oder REST-API – alle drei Wege decken denselben Funktionsumfang ab, keine Funktion ist der Oberfläche vorbehalten
- Lizenzmodell: die komplette Software ist quelloffen (AGPLv3) und ohne Abo voll funktionsfähig – ein kostenpflichtiges Support-Abo ist optional und ändert am Funktionsumfang nichts, nur am Zugriff auf das Enterprise-Repository und den Herstellersupport
- Entwickelt von Proxmox Server Solutions GmbH (Wien) – seit 2008 am Markt, damit kein junges Projekt, sondern eine etablierte, kontinuierlich weiterentwickelte Plattform

Architektur: Datacenter → Node → Guest



- **Datacenter:** die logische Klammer über alle verbundenen Nodes (auch als Einzel-Node nutzbar) – in der Web-GUI die oberste Ansicht, von der aus Storage, Berechtigungen und Cluster-weite Einstellungen verwaltet werden
- **Node:** ein physischer oder virtueller Proxmox-Host mit eigenem Storage und Netzwerk – jeder Node bringt seine eigene Konfiguration unter `/etc/pve` mit, die im Cluster-Verbund automatisch zwischen allen Nodes synchronisiert wird
- **Guest:** die einzelne VM oder der einzelne Container auf einem Node – identifiziert über eine eindeutige VMID, die clusterweit nur einmal vergeben wird
- Diese dreistufige Hierarchie bleibt gleich, egal ob ein einzelner Mini-Server oder ein Cluster mit einem Dutzend Nodes betrieben wird – sie wächst einfach mit

KVM vs. LXC



KVM

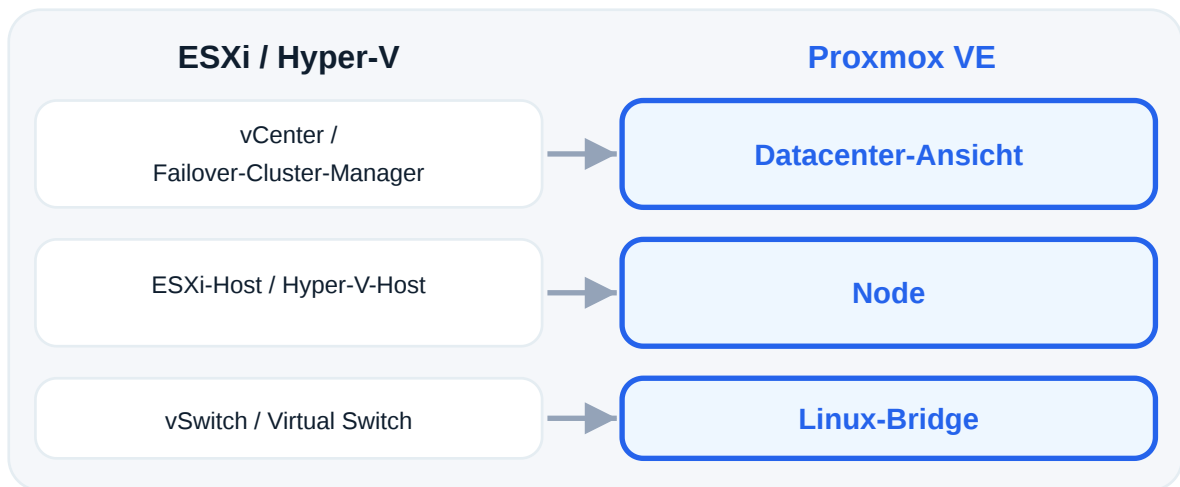
Vollwertige VM mit eigenem Kernel – jedes Gast-Betriebssystem, volle Isolation, mehr Overhead durch Hardware-Emulation

LXC

Container teilt sich den Host-Kernel – sehr schlank und schnell, startet in Sekunden, aber nur für Linux-Gäste geeignet

Faustregel: Windows und alles Gast-OS-Unabhängige → KVM. Schlanke Linux-Dienste, bei denen Performance und schneller Start zählen (z. B. ein einzelner Webserver oder Reverse Proxy) → LXC. Im Zweifel ist KVM die sicherere Wahl, weil die Isolation vollständiger ist – ein Kernel-Exploit im Container kann theoretisch den Host betreffen, bei einer VM nicht.

Denkweise-Umstieg von ESXi/Hyper-V



Praktisch heißt das: Firewall-Regeln, Netzwerkkonfiguration, sogar der Cluster-Zustand liegen als lesbare Textdateien unter `/etc/pve` – im Zweifel kannst du direkt nachsehen, statt einer Oberfläche vertrauen zu müssen. Das erleichtert auch die Fehlersuche erheblich: ein `cat` auf die passende Datei ersetzt oft mehrere Klicks durch verschachtelte Menüs.

ESXi/vCenter

vCenter → Datacenter-Ansicht · ESXi-Host → Node
· vSwitch → Linux-Bridge · Datastore → Storage

Hyper-V

Failover-Cluster-Manager → Datacenter-Ansicht ·
Hyper-V-Host → Node · Virtual Switch → Linux-
Bridge · CSV → Storage

Der größte Unterschied ist keine Funktion, sondern die Haltung: Proxmox zeigt dir das darunterliegende Linux, statt es hinter einer Appliance zu verstecken. Wer bisher nie eine Kommandozeile auf dem Hypervisor gebraucht hat, wird sie bei Proxmox zumindest gelegentlich sehen – meist aber nur zur Fehlersuche, der Alltag läuft über die Web-GUI.

Web-GUI, CLI & API

qm – VM-Verwaltung

```
qm start 100
```

Starten, Stoppen, Migrieren, Konfigurieren einzelner VMs per ID

pct – Container-Verwaltung

```
pct enter 101
```

Analoges Werkzeug zu `qm`, aber für LXC-Container – hier direkt eine Shell im Container öffnen

pvesh – REST-API-Zugriff

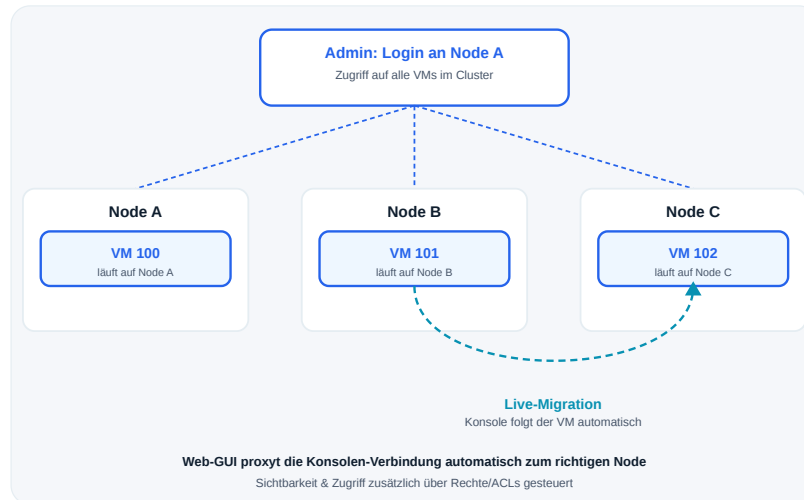
```
pvesh get /nodes/pve1/qemu
```

Liest/schreibt direkt über die REST-API – Basis für eigene Skripte, Terraform- oder Ansible-Anbindung

- Web-GUI: beste Wahl für den Alltag – Übersicht, Konsole, Logs und Konfiguration an einem Ort, erreichbar unter `https://<IP>:8006`
- CLI direkt auf dem Node per SSH – am schnellsten für einzelne, gezielte Befehle oder wenn die Web-GUI mal nicht erreichbar ist
- Für Automatisierung: API-Token statt Passwort verwenden, damit Skripte nicht mit vollen Admin-Zugangsdaten hantieren – Tokens lassen sich pro Anwendungsfall einzeln anlegen und widerrufen, ohne das Hauptpasswort zu ändern
- Die REST-API ist dieselbe Grundlage, auf der auch Terraform-Provider, Ansible-Module und Backup-Tools von Drittanbietern aufsetzen

Alle drei Zugangswege sprechen intern dieselbe Proxmox-API – nichts, was die Oberfläche kann, bleibt der Kommandozeile verwehrt, und umgekehrt. Das ist ein bewusster Architekturentscheid: Jeder Klick in der Web-GUI löst im Hintergrund denselben API-Aufruf aus, den auch ein eigenes Skript absetzen könnte.

Cluster-weiter Konsolenzugriff



Ein Login an einem einzigen Node genügt für den Zugriff auf alle VMs im gesamten Cluster – genau wie bei vCenter, egal auf welchem Node die jeweilige VM tatsächlich läuft.

- noVNC- bzw. SPICE-Konsole funktioniert für jede VM auf jedem Node – die Web-GUI proxyt die Konsolen-Verbindung automatisch zum richtigen Node im Hintergrund
- Nach einer Live-Migration bleibt der Konsolenzugriff erhalten – die VM zieht auf einen anderen Node um, und der Zugriff „folgt“ ihr automatisch, ohne dass eine neue Verbindung aufgebaut werden muss
- Voraussetzung: alle beteiligten Nodes müssen im selben Cluster sein, Corosync bzw. das Cluster-Netz muss funktionieren, und der Node, auf dem die VM tatsächlich läuft, muss erreichbar sein
- Rechte/ACLs steuern zusätzlich, wer welche VMs überhaupt sehen und darauf zugreifen darf – wichtig bei Mandantentrennung, etwa im MSP-Betrieb mit mehreren Kunden auf demselben Cluster
- Fällt der Node aus, an dem man gerade eingeloggt ist, meldet man sich einfach an einem anderen Node an – die Web-GUI läuft identisch auf jedem Node im Cluster
- Die Mechanik dahinter (Proxy-Prinzip, Cluster-weite Sichtbarkeit) wird in [Modul 7 – Cluster & Quorum](#) vertieft

Nested Virtualization

Eine VM in einer VM betreiben – nützlich für Testlabore und Migrationsproben, ohne echte Hardware zu blockieren.

- Voraussetzung: Intel VT-x/EPT bzw. AMD-V/NPT auf Host und CPU-Typ der äußeren VM (z. B. `host` oder ein moderner `x86-64-v3`) – ohne diese Weitergabe verweigert die innere Proxmox-Installation den Start eigener VMs
- Aktivierung über die VM-Optionen: CPU-Flag „`vmx`“ (Intel) bzw. „`svm`“ (AMD) muss in der äußeren VM sichtbar sein, meist per CPU-Typ „`host`“ oder gezieltem Flag automatisch der Fall
- Typischer Einsatz: einen kompletten Proxmox-Cluster testweise in mehreren VMs nachbauen, bevor echte Hardware angefasst wird – genau das nutzen auch die Beispiele in diesem Lernbereich
- Auch nützlich für Schulungen und Vorführungen – ein ganzes Mehr-Node-Setup mit Cluster und HA läuft auf einem einzigen Laptop oder Mini-PC
- Performance-Abzug gegenüber echter Hardware einplanen (grob 10–20 % je nach Workload) – für Produktivbetrieb ungeeignet, aber für Lernen, Tests und Migrationsproben ideal

Das Wichtigste

Proxmox VE ist im Kern kein neues Produkt, sondern Debian plus KVM/QEMU plus LXC, zusammengehalten von einer durchdachten Weboberfläche – wer Linux kennt, hat schon die halbe Lernkurve hinter sich.

- Zwei Virtualisierungsarten in einer Oberfläche: KVM für vollwertige VMs, LXC für schlanke Linux-Container
- Klare Hierarchie: Datacenter → Node → Guest, auch als Einzel-Node-Setup nutzbar
- Von ESXi/Hyper-V übersetzt sich fast jedes Konzept eins zu eins – die Umstellung ist mehr Haltung als Funktion
- Gleicher Funktionsumfang über GUI, `qm / pct` oder die `pvesh`-API – wichtig für spätere Automatisierung
- Nested Virtualization eignet sich für Labs und Schulungen, nicht für Produktivbetrieb

M2 · Hardwareauswahl & CPU

Die richtige Basis, bevor die erste VM überhaupt startet

Server, Mini-PC oder selbstgebaut?

Server

IPMI/BMC, Redundanz, Hersteller-Support –
Standard für Produktivbetrieb

Mini-PC

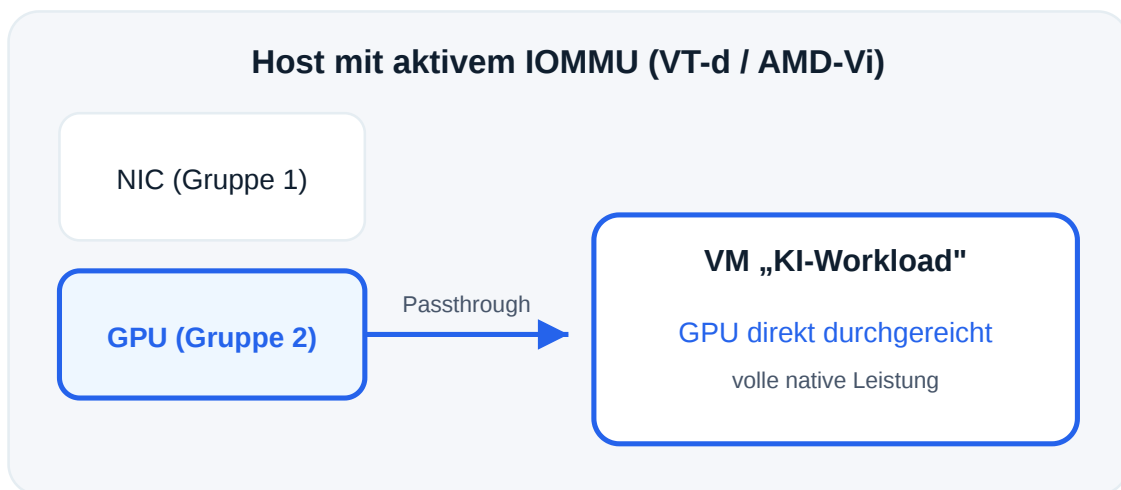
Leise, günstig, für Labs oder kleine Homeoffice-
Setups – kaum Redundanz

Selbstgebaut

Volle Kontrolle über Komponenten, aber Support-Risiko und Kompatibilitätsprüfung liegen bei dir

Wichtig: Proxmox VE läuft auf x86-64-Hardware. Kein Apple Silicon – seit Version 9.2 (August 2026) gibt es ersten Arm64-Support für ausgewählte Serverplattformen, das bleibt aber vorerst die Ausnahme. Für ein erstes Testlabor reicht auch gebrauchte Business-Hardware (z. B. ein ausrangierter Dell/HP/Lenovo-Tower-Server) völlig aus – die Kernanforderungen (VT-x/AMD-V, genug RAM) erfüllen auch mehrere Jahre alte Modelle.

CPU-Voraussetzungen



- Virtualisierungserweiterung zwingend: Intel VT-x oder AMD-V – ohne sie startet Proxmox VE gar nicht erst, die meisten Server- und Desktop-CPU's der letzten 15 Jahre bringen sie aber mit
- Für GPU-/Geräte-Passthrough zusätzlich IOMMU nötig: Intel VT-d bzw. AMD-Vi, im BIOS/UEFI aktivieren (Bezeichnung variiert je Hersteller, z. B. „Intel VT for Directed I/O“)
- Ohne IOMMU kein sauberes Durchreichen einzelner PCIe-Geräte an eine VM – das Gerät bliebe entweder dem Host vorbehalten oder müsste per Software emuliert werden, was Performance kostet
- IOMMU-Gruppierung prüfen: Geräte, die in derselben Gruppe stecken, lassen sich nur gemeinsam durchreichen – bei ungünstigem Mainboard-Layout betrifft das mitunter mehr Hardware als gewünscht
- Praxisrelevant vor allem für KI-/ML-Workloads (Inferenz, Training, lokale LLMs): **GPU-Passthrough** reicht eine physische GPU komplett und exklusiv an eine einzige VM durch – volle native Leistung, aber die Karte steht anderen VMs dann nicht mehr zur Verfügung. **vGPU** (GPU-Sharing) teilt eine GPU dagegen auf mehrere VMs auf, ist aber herstellerabhängig und meist nur mit Profi-/Datacenter-Karten samt zusätzlicher Lizenz (z. B. NVIDIA vGPU) möglich – Consumer-GPUs unterstützen kein vGPU-Sharing, lassen sich aber weiterhin per klassischem Passthrough an eine VM durchreichen

Intel Xeon vs. AMD EPYC

Intel Xeon

Breite Kompatibilität, viele Mainboard-Optionen, etablierte Enterprise-Features

AMD EPYC

Sehr hohe Kernzahlen und Speicherbandbreite pro Sockel, oft attraktiveres Preis-Leistungs-Verhältnis

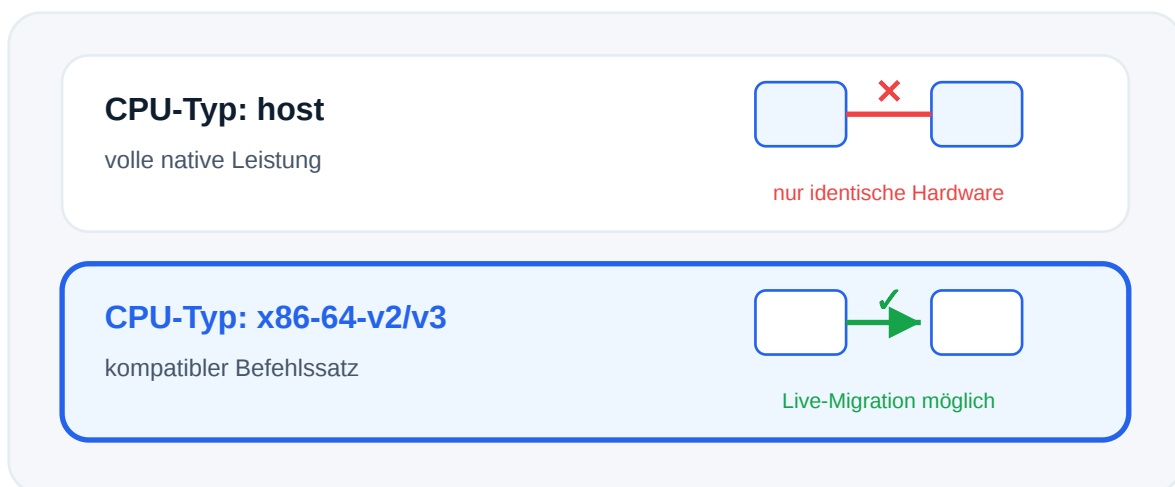
Beide funktionieren zuverlässig mit Proxmox – die Entscheidung ist meist eine Frage von Budget, Kernzahl-Bedarf und vorhandenem Server-Ökosystem. Für Live-Migration innerhalb eines Clusters gilt: Intel- und AMD-CPU's lassen sich nicht mischen (unterschiedliche Befehlssatz-Familien), innerhalb einer Herstellerlinie ist ein CPU-Typ wie `x86-64-v3` aber auch über mehrere Prozessorgenerationen hinweg migrationsfähig.

ECC-RAM, PCIe-Lanes & NUMA

Ein einzelnes, unkorrigiertes Bit-Kippen im RAM kann bei ZFS zu einem stillen Datenfehler führen, der sich unbemerkt über Snapshots und Replikation fortpflanzt – ECC ist deshalb keine Kür, sondern Grundabsicherung.

- ECC-RAM korrigiert Bitfehler selbstständig – für Produktivbetrieb dringend empfohlen, gerade bei ZFS, das große Teile des RAM als Cache (ARC) nutzt und Fehler dort sonst ungeprüft in den Pool übernehmen würde
- Genug PCIe-Lanes einplanen, wenn mehrere Karten (NIC, HBA, GPU) gleichzeitig durchgereicht werden sollen – Consumer-Mainboards haben oft nur 20–24 Lanes, Server-Plattformen dagegen 64–128 oder mehr
- NUMA beachten: VM-Speicher und -Kerne möglichst auf einem NUMA-Knoten halten, sonst Leistungseinbußen durch Zugriffe über den Interconnect zwischen den Sockeln (spürbar vor allem bei speicherintensiven Workloads wie Datenbanken)
- Bei Zwei-Sockel-Systemen: VM-Größe nach Möglichkeit an die Kern-/Speichermenge eines einzelnen NUMA-Knotens anpassen – Proxmox zeigt die NUMA-Topologie unter Node → System → Hardware an
- Faustregel für die RAM-Planung: genug Puffer für ZFS-ARC einrechnen (Standard bis zu 50 % des RAM) und nicht den kompletten Host-Speicher an VMs verplanen

CPU-Typ in der VM



Der CPU-Typ „host“ gibt der VM alle Fähigkeiten der echten CPU weiter – schnellster Betrieb, aber Live-Migration nur zwischen identischer Hardware, weil das Gast-Betriebssystem sich exakt auf die vorhandenen Befehlssatzerweiterungen einstellt.

- x86-64-v2 / x86-64-v3 : gemeinsamer Befehlssatz-Kompromiss (kleinster Nenner moderner CPUs) – Live-Migration auch zwischen unterschiedlichen CPU-Generationen und sogar zwischen Intel und AMD möglich
- Faustregel: gemischter Cluster oder unklare zukünftige Hardware → x86-64-v2/v3 , homogener Cluster mit Fokus auf Leistung → host
- Der CPU-Typ lässt sich pro VM in den Hardware-Optionen ändern, wirkt aber erst nach einem Neustart der VM – ein nachträglicher Wechsel ist also möglich, sollte aber getestet werden, bevor er produktiv nötig wird

Redundanz einplanen

- Redundante Netzteile gegen Einzelausfälle – idealerweise an getrennte Stromkreise (im Idealfall an getrennte USV-Systeme) angeschlossen
- IPMI/BMC für Fernwartung und Neustart, auch wenn das Betriebssystem nicht mehr reagiert – erspart nächtliche Anfahrten ins Rechenzentrum und ermöglicht Remote-Konsolenzugriff schon vor dem Betriebssystem-Boot
- Redundante NICs, idealerweise über getrennte Switches, für Netzwerk- und Corosync-Ausfallsicherheit – realisiert über einen Bond (siehe Modul 4 und 6)
- Bei Clustern: eine dedizierte, niedrig-latente Netzwerkverbindung nur für Corosync einplanen (mehr dazu in Modul 7) – Corosync reagiert empfindlich auf Netzwerk-Jitter, geteilte Bandbreite mit VM-Traffic ist ein häufiger Grund für ungewollte Fencing-Ereignisse
- Bei Storage: mindestens zwei Netzwerke UND, wo Budget erlaubt, redundante Storage-Controller/HBAs einplanen – ein einzelner HBA-Ausfall sollte nicht gleich den ganzen Node lahmlegen

Das Wichtigste

Die teuersten Hardware-Fehler passieren vor dem Kauf: fehlendes IOMMU, zu wenig PCIe-Lanes oder ECC vergessen lassen sich hinterher nicht mit einem Software-Update reparieren.

- x86-64 ist Pflicht, VT-x/AMD-V ebenso – IOMMU zusätzlich für Passthrough (Arm64 bleibt vorerst die Ausnahme)
- Intel Xeon und AMD EPYC funktionieren beide zuverlässig, die Wahl ist meist Budget- und Kernzahl-getrieben, ein Mischbetrieb im selben Cluster ist bei Live-Migration aber nicht möglich
- ECC-RAM, ausreichend PCIe-Lanes und NUMA-Bewusstsein zahlen sich im Betrieb aus – besonders bei ZFS und speicherintensiven VMs
- CPU-Typ „host“ vs. „x86-64-v2/v3“ entscheidet über die Live-Migrations-Flexibilität – nachträglich änderbar, aber vorher planen erspart spätere Neustarts
- Redundanz (Netzteile, IPMI/BMC, NICs, Storage-Controller) gehört von Anfang an in die Planung, nicht in die Nachrüstung

M3 · Storage & Filesystem

Die Entscheidung, die man später nur schwer rückgängig macht

Storage-Optionen im Überblick

ZFS

Checksummen, Snapshots, Replikation eingebaut – Proxmox-Favorit für lokalen Storage

LVM-thin

Schlank, schnell, gute Snapshot-Unterstützung – kein Checksumming

ext4

Einfach und robust, aber ohne native Snapshots für VM-Disks

Ceph

Verteilter Speicher über mehrere Nodes – echtes Shared Storage im Cluster

Diese vier Optionen decken die allermeisten Proxmox-Installationen ab. Für den Anfang gilt: ZFS ist die im Ökosystem am besten integrierte Wahl (Snapshots, Replikation und Backup-Verifizierung greifen direkt ineinander) – LVM-thin und ext4 sind eher Optionen für sehr einfache Setups oder wenn ein vorhandener Hardware-RAID-Verbund weiterverwendet werden muss.

RAID-Level im Vergleich

Klassisches RAID (Hardware-Controller oder Linux- mdadm) kennt kein Checksumming gegen stille Bitfehler. ZFS bildet dieselben Redundanzkonzepte nach – Mirror $\approx$ RAID 1, RAID-Z $\approx$ RAID 5, RAID-Z2 $\approx$ RAID 6 – aber komplett in Software, mit Prüfsummen auf jedem Block und automatischer Selbstheilung bei erkannten Fehlern.

RAID 0 (Striping)

Volle Kapazität und Performance, aber keine Redundanz – ein einziger Plattenausfall bedeutet Totalverlust

RAID 1 (Mirror)

Zwei identische Kopien, einfachste Redundanz, nur 50 % Nutzkapazität, schnelle Wiederherstellung

RAID 5

Parität über alle Platten verteilt, verkraftet einen Ausfall, Write-Penalty durch Paritätsberechnung, riskanter Rebuild bei sehr großen Platten

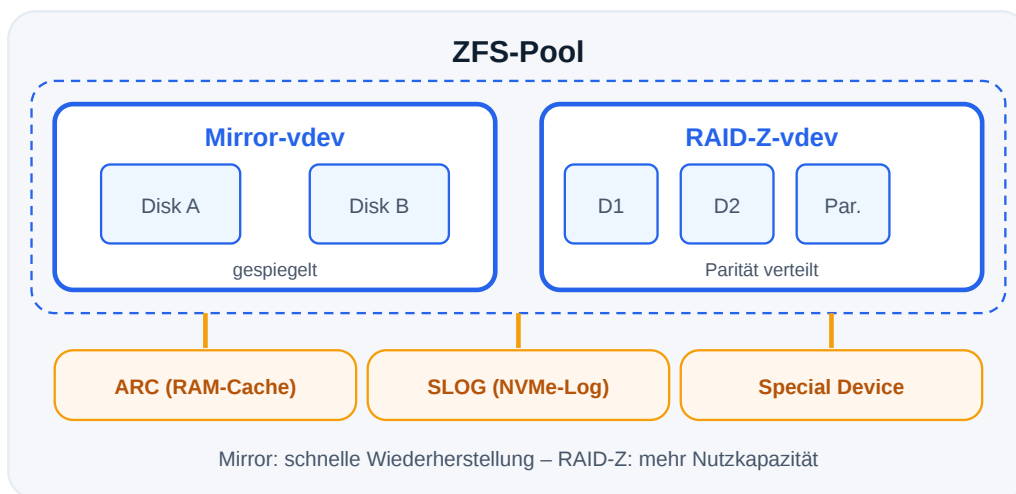
RAID 6

Zwei Paritäten, verkraftet zwei gleichzeitige Ausfälle, mehr nutzbarer Verlust an Kapazität als RAID 5

RAID 10

Mirror + Stripe kombiniert, sehr gute Performance und Redundanz, aber wie RAID 1 nur die Hälfte der Rohkapazität nutzbar

ZFS im Detail



- **Mirror:** einfache Spiegelung (entspricht RAID 1), schnell wiederhergestellt, weniger Nutzkapazität
- **RAID-Z:** parity-basiert (entspricht RAID 5), mehr Nutzkapazität, längerer Resilver bei Plattenausfall; **RAID-Z2** verkraftet wie RAID 6 zwei gleichzeitige Ausfälle
- **ARC:** RAM-Cache für Lesezugriffe – ZFS ist speicherhungrig, das ist gewollt, denn jeder RAM-Cache-Treffer erspart einen langsamen Plattenzugriff
- **SLOG:** schnelles Log-Device (z. B. NVMe) beschleunigt synchrone Schreibzugriffe (z. B. Datenbanken) – ohne SLOG landet dieses Log auf dem regulären, langsameren Pool
- **Special Device:** lagert Metadaten und optional kleine Dateien auf schnellem Storage aus, beschleunigt Verzeichnis-Operationen und viele kleine Dateien spürbar

Hohe I/O-Anforderungen (z. B. Datenbanken)

Datenbank-VMs erzeugen viele kleine, häufig synchrone Schreibzugriffe (jede Transaktion wartet auf Bestätigung) – das ist ein grundlegend anderes Lastprofil als große, sequenzielle Datei-Kopien und stellt eigene Anforderungen an Storage und VM-Konfiguration.

- NVMe statt SATA/SAS-SSD für die VM-Disk einplanen, wenn die Anwendung latenzempfindlich ist – die geringere Zugriffszeit wirkt sich bei vielen kleinen IOPS deutlich stärker aus als bei sequenziellem Durchsatz
- Cache-Modus der VM-Disk auf `none` stellen (Proxmox-Standard) – vermeidet doppeltes Caching zwischen Host und Gast, das bei Datenbanken eher zu Inkonsistenzrisiken als zu Performance-Gewinn führt
- Als Async-IO-Modus `io_uring` nutzen (moderner Standard) statt `threads` – geringerer CPU-Overhead pro I/O-Operation bei hoher Parallelität
- SLOG-Device (siehe ZFS im Detail) lohnt sich hier besonders: es beschleunigt genau die synchronen Schreibzugriffe, auf die Datenbank-Commits warten
- Bei sehr hohen Ansprüchen: eigenen Storage-Pool nur für I/O-intensive VMs reservieren, damit ein „lauter Nachbar“ (z. B. ein Backup-Job) nicht die Datenbank-Latenz verschlechtert

Fallstrick: RAID-Controller

ZFS und Ceph wollen direkten Zugriff auf die Platten – ein Hardware-RAID-Controller darunter stört mehr, als er hilft.

- Controller in den HBA- bzw. IT-Modus versetzen (Firmware-Flash nötig, falls der Controller werkseitig nur RAID-Modus anbietet, z. B. bei vielen Dell-PERC- oder HP-Smart-Array-Controllern)
- Hardware-RAID verschleiert den Plattenzustand vor ZFS/Ceph – Checksumming und Selbstheilung greifen dann nicht mehr richtig, weil ZFS die einzelnen physischen Platten gar nicht mehr sieht, sondern nur noch das vom Controller zusammengefasste virtuelle Laufwerk
- Erkennungsmerkmal im Zweifel: `lsblk` oder `smartctl` sollten die einzelnen physischen Platten einzeln auflisten – erscheint stattdessen nur ein einziges großes „virtuelles“ Laufwerk, steckt wahrscheinlich noch Hardware-RAID dazwischen
- Wer zwingend mit vorhandenem Hardware-RAID-Controller leben muss, sollte auf ZFS/Ceph verzichten und stattdessen ext4 oder LVM-thin auf dem vom Controller bereitgestellten Verbund nutzen

Local vs. Shared Storage

Local Storage

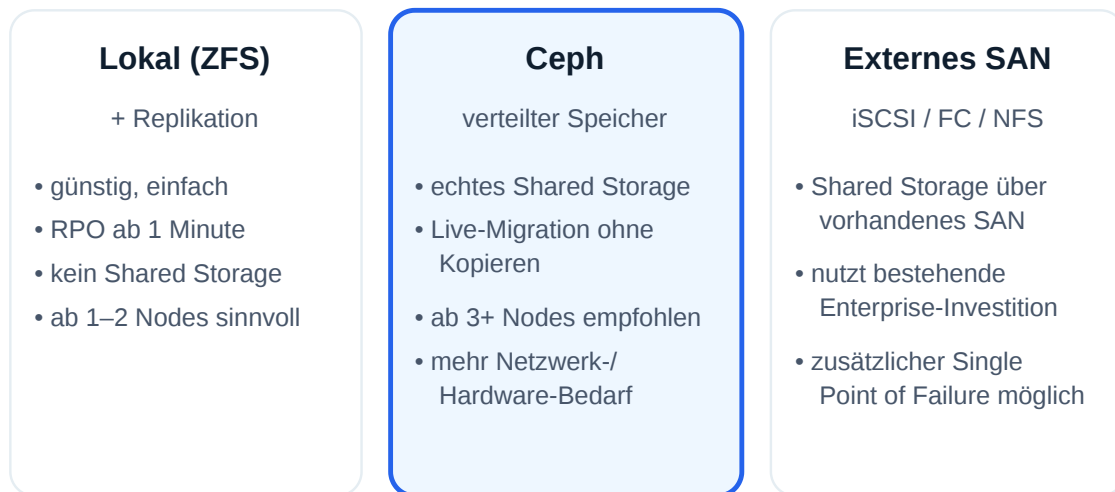
Günstig und einfach – für HA braucht es zusätzlich Replikation, Failover kostet die letzten Minuten Daten (RPO)

Shared Storage

Ceph oder SAN – Live-Migration ohne Datenkopie, Grundlage für nahtlose Hochverfügbarkeit

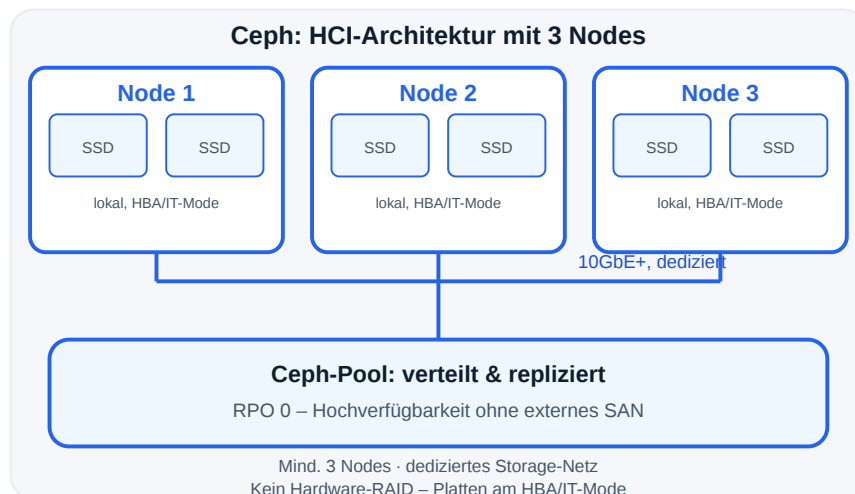
Der praktische Unterschied zeigt sich bei der Live-Migration: Bei lokalem Storage muss die komplette VM-Disk zum Ziel-Node kopiert werden (dauert je nach Größe und Netzwerk spürbar), bei Shared Storage sieht der Ziel-Node dieselben Daten bereits – nur Arbeitsspeicher und CPU-Zustand wandern, die Migration ist entsprechend schneller und unterbrechungsärmer.

Entscheidungsvergleich



Die drei Wege unterscheiden sich vor allem im Verhältnis von Aufwand zu Ausfallsicherheit: lokales ZFS mit Replikation ist am schnellsten aufgesetzt, akzeptiert dafür ein kleines RPO im Failover-Fall (siehe Modul 9); Ceph braucht mehr Planung und mindestens drei Nodes, liefert dafür echtes Shared Storage ohne Datenverlust bei einem Node-Ausfall; ein bestehendes SAN nutzt vorhandene Investitionen, bringt aber einen zusätzlichen Single-Point-of-Failure außerhalb des Proxmox-Clusters mit, sofern das SAN selbst nicht redundant ausgelegt ist.

HCI: vSAN-Ersatz mit Ceph



Ceph ist das direkteste vSAN-Äquivalent in Proxmox: verteilter Storage aus lokalen Node-Platten, RPO 0, Hochverfügbarkeit ohne externes SAN.

- **Voraussetzungen:** mindestens 3 Nodes, dediziertes 10GbE+ (oder schnelleres) Storage-Netz, Platten am HBA/IT-Mode statt Hardware-RAID, SSD/NVMe empfohlen
- Abgrenzung zu **ZFS-Replikation**: kein echtes Shared Storage, da nur periodisch repliziert wird – dadurch RPO größer 0 statt RPO 0 wie bei Ceph
- Abgrenzung zu **Linstor/DRBD**: Block-Replikation zwischen Nodes, ein anderer technischer Ansatz als der verteilte Ceph-Storage
- Klarstellung: **VMware vSAN** selbst läuft nicht unter Proxmox – wohl aber das dahinterliegende HCI-Prinzip (hyperkonvergente, verteilte Storage-Architektur), das Ceph umsetzt

Empfehlung nach Node-Anzahl & Budget

- 1–2 Nodes, kleines Budget: lokales ZFS mit Replikation – pragmatisch, ausreichend für die meisten KMU, RPO im Minutenbereich ist meist akzeptabel
- 3+ Nodes, Wachstum absehbar: Ceph – mehr Aufwand beim Aufbau (eigenes Storage-Netz empfohlen), dafür sauberes Shared Storage und Kapazität, die sich durch weitere Nodes einfach erweitern lässt
- Bereits vorhandenes SAN im Haus: anbinden statt neu investieren, sofern Performance und Redundanz passen – SAN-Anbindung meist per iSCSI oder NFS, abhängig vom Hersteller
- Faustregel für den Einstieg: erst mit lokalem ZFS produktiv gehen und Erfahrung sammeln, Ceph erst einführen, wenn die Node-Anzahl und der Bedarf an echtem Shared Storage tatsächlich da sind – ein vorschneller Ceph-Aufbau mit zu wenig Nodes oder zu langsamem Netz enttäuscht häufiger, als er hilft
- Für das KMU-Referenz-Szenario dieses Lernbereichs (1–2 Nodes, 4–6 VMs, siehe Modul 0) ist damit die Antwort meist eindeutig: lokales ZFS mit Replikation, nicht Ceph – die Datenmenge und Node-Anzahl rechtfertigen den zusätzlichen Aufwand in aller Regel noch nicht

Snapshots & das Wichtigste

Fallstrick: LVM-thick kann keine Snapshots von VM-Disks – für Backups und schnelle Rücksicherungen ZFS, LVM-thin oder Ceph bevorzugen.

Ein Snapshot ist kein Backup-Ersatz: er liegt auf demselben Storage bzw. Pool wie die VM selbst und schützt daher nicht vor einem Hardware- oder Pool-Ausfall – für echte Datensicherung bleibt ein separates Backup auf einem anderen Ziel notwendig (siehe Backup-Modul).

- ZFS und Ceph: Snapshots + eingebaute Redundanz/Selbstheilung – ein Snapshot ist in Sekunden erstellt und kostet zunächst keinen zusätzlichen Speicherplatz (nur die seither geänderten Blöcke wachsen)
- RAID-Controller weg – ZFS/Ceph brauchen direkten Plattenzugriff über HBA/IT-Mode, sonst funktionieren Checksumming und Selbstheilung nicht wie vorgesehen
- Bei hohen I/O-Anforderungen (Datenbanken) zusätzlich NVMe, `cache=none` und ggf. ein SLOG-Device einplanen
- Storage-Wahl richtet sich nach Node-Anzahl, Budget und HA-Anspruch, nicht nach Gewohnheit – was beim bisherigen ESXi/Hyper-V-Setup funktioniert hat, ist nicht automatisch die richtige Wahl für Proxmox
- Snapshot-Fähigkeit hängt am Storage-Typ: **qcow2**, **ZFS** und **Ceph** unterstützen native Snapshots von VM-Disks, **LVM-thick** ausdrücklich nicht (LVM-thin dagegen schon)
- Anders als bei VMware: Proxmox-Snapshots sind kein einheitliches Hypervisor-Format, sondern hängen vom gewählten Storage-Backend ab (z. B. ZFS-COW-Blocksnapshot oder Ceph-RBD-Snapshot statt VMFS-Delta-Datei)

M4 · Installation & Ersteinrichtung

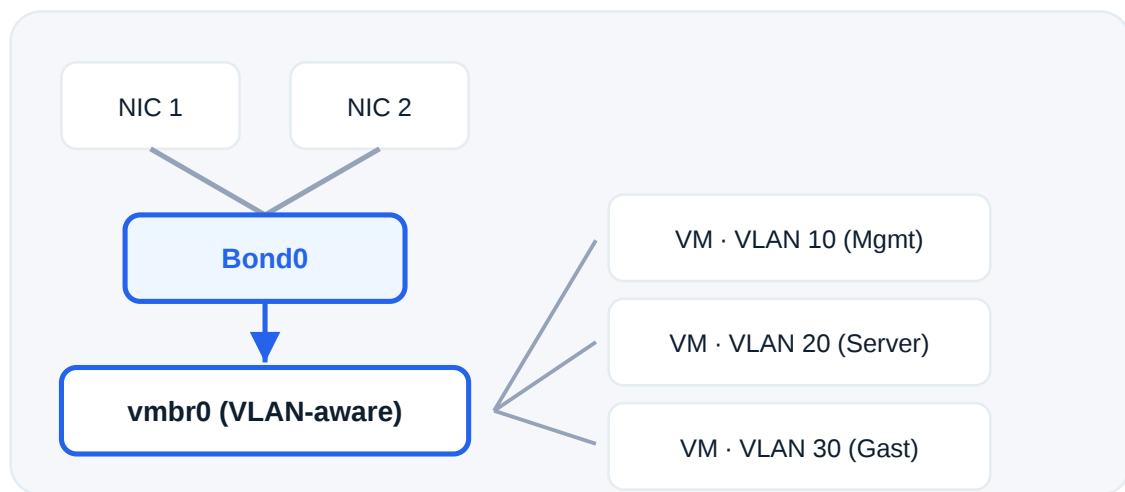
Vom ISO-Image zum abgesicherten, einsatzbereiten Host

Installationsablauf

Von ISO-Boot bis einsatzbareitem Host vergehen in der Praxis oft nur 10–15 Minuten – der Assistent übernimmt Partitionierung, Storage-Layout und Basis-Netzwerk in einem Rutsch.

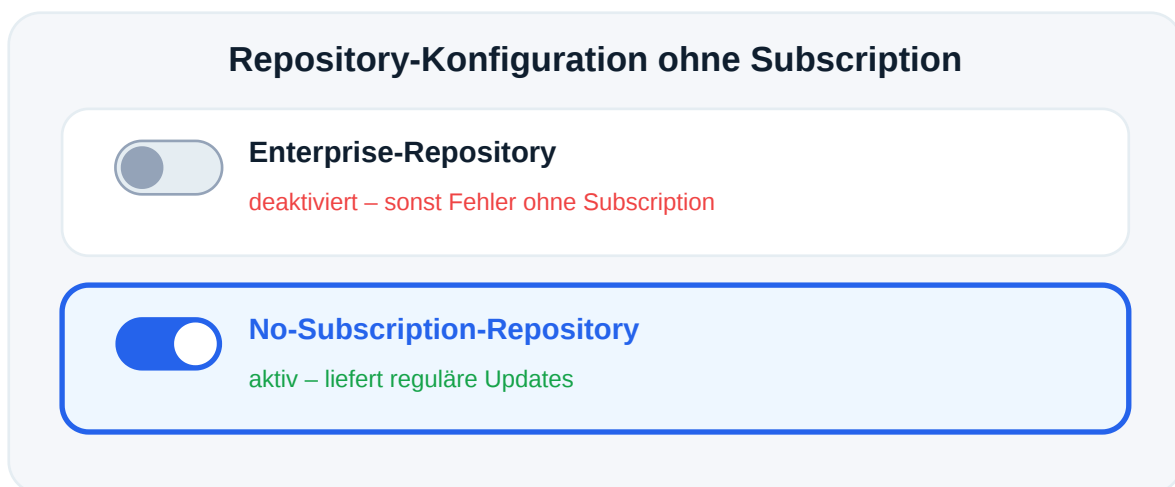
- ISO von proxmox.com/downloads herunterladen und auf einen USB-Stick schreiben – unter Windows z. B. mit **Rufus** (DD-Modus wählen, nicht ISO-Modus, sonst startet der Stick nicht richtig), unter Linux/macOS reicht `dd if=proxmox-ve.iso of=/dev/sdX bs=4M status=progress` oder das grafische balenaEtcher
- Zum Ausprobieren ohne echte Hardware: Proxmox VE lässt sich auch als verschachtelte VM in einer bestehenden Virtualisierungsumgebung installieren (z. B. in VirtualBox, VMware Workstation oder sogar in einer anderen Proxmox-VM mit aktivierter Nested Virtualization, siehe Modul 1) – ideal für ein risikofreies Testlabor, bevor echte Server angefasst werden
- Installation per ISO, geführter Assistent für Storage-Layout (ZFS, LVM oder ext4) und Netzwerk
- Boot-Modus UEFI empfohlen, dafür läuft jede VM standardmäßig mit OVMF (UEFI-Firmware für Gäste) – Voraussetzung u. a. für Windows-11-Gäste mit TPM
- Nach der Installation: Web-GUI sofort unter `https://<IP>:8006` erreichbar, kein Zusatzschritt nötig
- Hostname und IP-Adresse vorab festlegen – nachträgliches Ändern ist möglich, aber im Cluster-Verbund unnötig fehleranfällig

Netzwerk-Grundkonfiguration



- **Bridge** (`vmbr0`): virtueller Switch, an den VMs und Container angeschlossen werden – Standard-Baustein jeder Proxmox-Installation
- **Bond**: bündelt mehrere physische NICs für Redundanz und/oder mehr Bandbreite – gängige Modi sind Active-Backup (einfach, ausfallsicher) und LACP (mehr Durchsatz, braucht passenden Switch)
- **VLAN**: eine VLAN-aware Bridge trennt Management-, Server- und Gast-Netz auf derselben Verkabelung – jede VM bekommt einfach ein Tag zugewiesen
- Praxis-Empfehlung: Management-Zugriff (Web-GUI, Corosync) auf ein eigenes VLAN legen, getrennt vom Produktiv-/Gastnetz

Repositories richtig einstellen



Frisch installiert ist standardmäßig das Enterprise-Repository aktiv – ohne gültige Subscription liefert das nur Fehlermeldungen bei Updates, weil der Zugriff serverseitig auf zahlende Abos beschränkt ist.

- No-Subscription-Repository aktivieren, wenn keine Subscription vorliegt – Einstellung über die Web-GUI (Node → Updates → Repositories) oder direkt in `/etc/apt/sources.list.d/`
- Enterprise-Repository deaktivieren, um störende Fehlermeldungen bei `apt update` zu vermeiden
- Mit Subscription: Enterprise-Repository für getestete, besonders stabile Updates nutzen – Pakete durchlaufen hier eine zusätzliche Testphase, bevor sie auch im No-Subscription-Kanal landen
- Ohne Subscription ist Proxmox VE trotzdem voll funktionsfähig – der Unterschied liegt ausschließlich bei Update-Kanal und Herstellersupport, nicht bei Features (siehe Modul 11)

Post-Install-Härtung

- SSH-Zugriff von Passwort auf Schlüssel umstellen, Root-Login per SSH-Passwort deaktivieren – Konfiguration in `/etc/ssh/sshd_config` (`PasswordAuthentication no`, `PermitRootLogin prohibit-password`)
- Integrierte Firewall für Management-Zugriffe (Web-GUI-Port 8006, SSH) auf bekannte Quell-IPs einschränken – Proxmox bringt eine eigene, cluster- und node-weite Firewall mit, die sich über die Web-GUI konfigurieren lässt
- Nur benötigte Repositories und Pakete installiert lassen – kleinere Angriffsfläche, weniger Update-Aufwand
- Automatische Sicherheitsupdates (`unattended-upgrades`) zumindest für kritische Patches erwägen – Abwägung zwischen Sicherheit und Kontrolle über den Update-Zeitpunkt, gerade im Cluster mit Rolling-Reboot-Strategie (Modul 5)
- Fail2ban oder die integrierte Proxmox-Firewall-Regel gegen Brute-Force auf die Web-GUI-Anmeldung ergänzen – Standard-Login-Versuche über das Internet sind keine Seltenheit, sobald Port 8006 erreichbar ist

DSGVO & NIS2: Compliance als Plattform-Kriterium

ANFORDERUNG	PROXMOX-ANTWORT
Datenhoheit / Serverstandort (DSGVO)	Self-hosted, eigenes Haus/EU
Löschkonzept & Aufbewahrung (DSGVO Art. 17)	PBS-Retention-Regeln (M13)
Techn./organis. Maßnahmen (DSGVO Art. 32)	Firewall, 2FA, ACLs (M4/M6)
Risikomanagement-Maßnahmen (NIS2)	Update-Mgmt, Segmentierung (M5/M6)
Business Continuity / Recovery (NIS2)	HA, Replikation, Backup (M8/M9/M13)
Zugriffskontrolle (NIS2 + DSGVO)	Rollen, ACLs, 2FA (M4)
Meldepflicht-Grundlage, Protokollierung (NIS2)	Audit-Log, Benachrichtigungen (M5)
Nachvollziehbarkeit / Lieferkette (NIS2)	Offene Config-Dateien (M1)
Kein Rechtsrat – reine technische Einordnung, welche Proxmox-Funktion auf welche Compliance-Anforderung einzahlt	

Reine Lerninhalte, keine Rechtsberatung: ob und in welchem Umfang NIS2 für das eigene Unternehmen gilt, hängt von Sektor, Größe und Umsatz ab (grobe Richtschnur: ab 50 Mitarbeitenden bzw. 10 Mio. € Jahresumsatz in bestimmten Sektoren) – im Zweifel spezialisierte Rechtsberatung hinzuziehen. Die DSGVO gilt dagegen unabhängig von der Unternehmensgröße für praktisch jedes Unternehmen, das personenbezogene Daten verarbeitet.

- DSGVO-Relevanz: Self-hosted-Betrieb im eigenen Haus gibt volle Kontrolle über den Serverstandort und den Zugriff auf personenbezogene Daten – ein Kriterium, das bei Cloud-Angeboten außerhalb der EU oft zusätzliche Vertragskonstrukte (Standardvertragsklauseln, Auftragsverarbeitung) nötig macht
- NIS2 (in Kraft seit Oktober 2024 EU-weit, deutsche Umsetzung über das NIS2UmsuCG) erweitert den Kreis betroffener Unternehmen deutlich über klassische kritische Infrastruktur hinaus – u. a. Risikomanagement-Maßnahmen, Meldepflichten bei Sicherheitsvorfällen und Nachweise zur Business Continuity gehören zu den Kernpflichten
- Viele NIS2-Anforderungen sind ohnehin gute Praxis, unabhängig von der Rechtslage: Netzwerksegmentierung (Modul 6), konsequentes Patch-Management (Modul 5), Zugriffskontrolle mit Rollen und 2FA (dieses Modul) und ein funktionierendes Backup-/Recovery-Konzept (Modul 13) zahlen gleichzeitig auf Compliance und auf ganz praktische Betriebssicherheit ein
- Ein handfester Vorteil gegenüber geschlossenen Appliances: die Proxmox-Konfiguration liegt als lesbare Textdateien unter `/etc/pve` vor (siehe Modul 1) – das erleichtert Nachweise und Audits erheblich, weil sich der tatsächliche Systemzustand jederzeit direkt einsehen und dokumentieren lässt, statt einer Blackbox vertrauen zu müssen

Zeitsynchronisation

Im Cluster ist eine korrekte Uhrzeit kein Detail: Corosync, Zertifikate und Log-Auswertung reagieren empfindlich auf Zeitdrift zwischen Nodes.

- NTP-Dienst auf allen Nodes gegen dieselbe Zeitquelle synchronisieren – Proxmox bringt standardmäßig `chrony` oder `systemd-timesyncd` mit, je nach Version
- Vor dem Cluster-Aufbau prüfen – nachträgliche Zeitsprünge können Corosync destabilisieren, da der Cluster-Kommunikationsprotokoll-Stack (Totem) auf enge Zeitfenster angewiesen ist
- Interne NTP-Quelle (z. B. die Firewall oder ein dedizierter Zeitserver im eigenen Netz) ist meist zuverlässiger und schneller als ein öffentlicher NTP-Pool über das Internet
- Zeitzone einheitlich auf allen Nodes setzen (meist UTC oder eine feste lokale Zeitzone) – uneinheitliche Zeitzonen erschweren später die Log-Korrelation zwischen Nodes

Zertifikate für die Web-GUI

- Standardmäßig ein selbstsigniertes Zertifikat – Browser-Warnung, aber verschlüsselt, für den ersten Zugriff direkt nach der Installation ausreichend
- ACME-Integration (z. B. Let's Encrypt) für ein vertrauenswürdigen Zertifikat ohne manuellen Aufwand, inklusive automatischer Erneuerung alle 90 Tage – Einrichtung unter Datacenter → ACME
- Wichtig bei mehreren Nodes: Zertifikat pro Node-Hostname einrichten – die Datacenter-Ansicht kann das zentral für den ganzen Cluster anstoßen, jeder Node braucht aber sein eigenes, auf seinen Hostnamen ausgestelltes Zertifikat
- Ohne öffentlich erreichbare Domain: ACME per DNS-Challenge nutzen, statt den Port 8006 ins Internet zu öffnen – unterstützt eine lange Liste an DNS-Providern über entsprechende Plugins
- Alternative für rein interne Umgebungen: eigene interne Zertifizierungsstelle (CA) nutzen und deren Root-Zertifikat einmalig auf den Verwaltungsrechnern als vertrauenswürdig hinterlegen

Benutzerverwaltung & das Wichtigste

Realms

Lokale Proxmox-Nutzer oder Anbindung an bestehendes AD/LDAP für zentrale Konten – kein Parallel-Nutzerverzeichnis pflegen

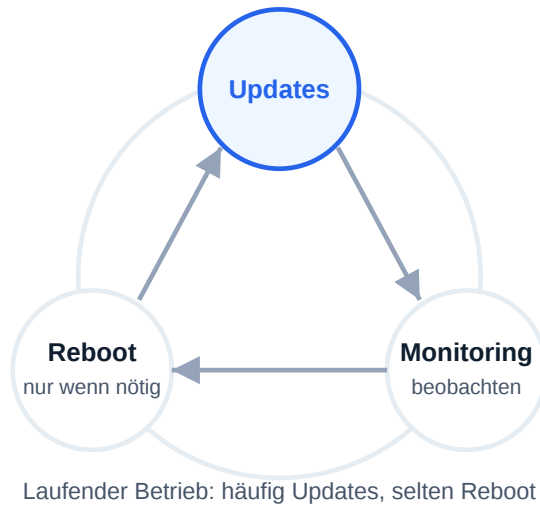
Rollen & ACLs

Feingranular je Pfad vergeben (z. B. nur eine VM verwalten) statt pauschaler Admin-Rechte für alle

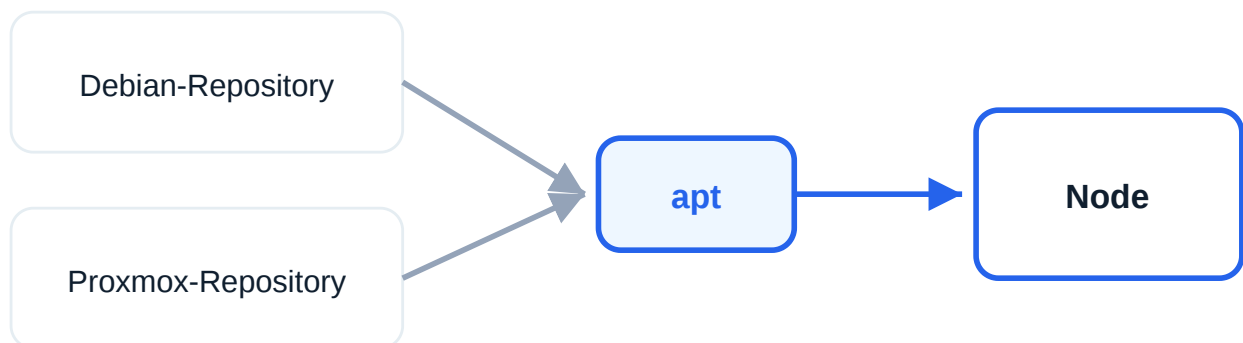
- Zwei-Faktor-Authentifizierung für die Web-GUI aktivieren, gerade für administrative Konten – unterstützt TOTP-Apps (z. B. Authy, Google Authenticator) und WebAuthn/Hardware-Keys
- Getrennte Konten statt geteiltem `root` -Zugang – jeder Admin bekommt sein eigenes Konto mit passender Rolle, das erleichtert später die Nachvollziehbarkeit im Audit-Log
- Merke: Repository-Wahl, Zeitsync, Härtung und Benutzerverwaltung gehören vor den Cluster-Aufbau, nicht danach – Nachrüsten im laufenden Cluster ist deutlich aufwendiger, weil Änderungen dann auf allen Nodes koordiniert nachgezogen werden müssen

M5 · Betrieb, Firmware & Monitoring

Der Alltag nach der Installation – Updates, Reboots, Hardware im Blick

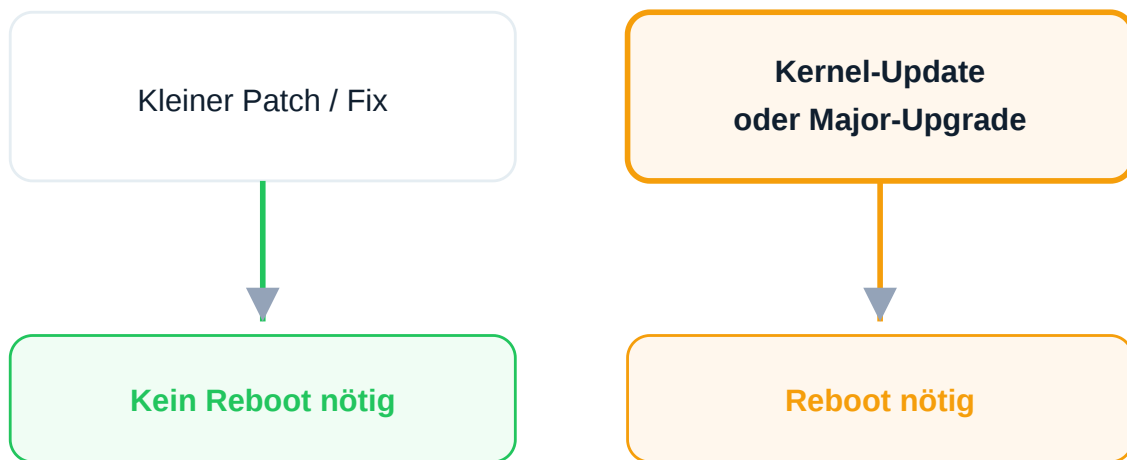


Update-Prinzip (Debian-Basis)



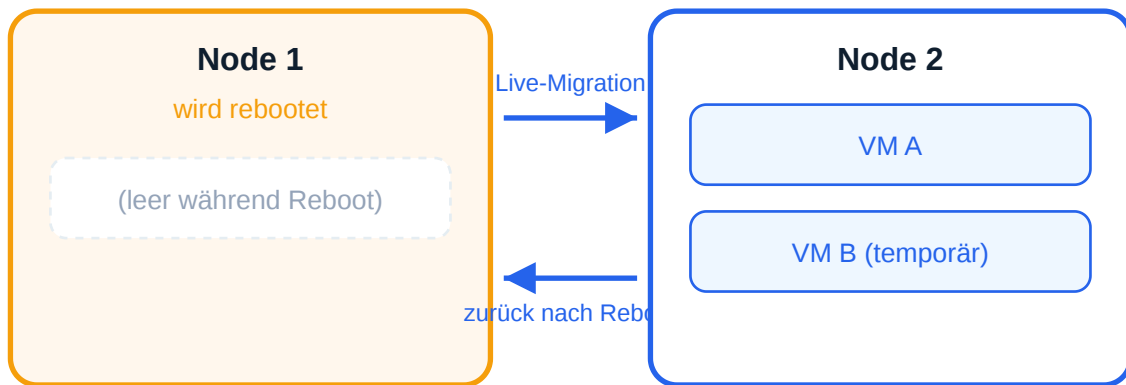
- Updates kommen aus zwei Quellen: dem Debian-Basissystem (Security- und Stable-Repository) und dem eigenen Proxmox-Repository für `pve-manager`, `qemu-server` und Co.
- Zwei Proxmox-Repo-Varianten: das **Enterprise-Repository** (Abo nötig, länger getestete Pakete, für Produktivsysteme empfohlen) und das kostenlose **No-Subscription-Repository** (aktueller, aber weniger lange im Feld erprobt)
- Kleine Fixes, vor allem Sicherheitspatches, können mehrmals pro Woche erscheinen
- Eingespielt wird klassisch per `apt update && apt full-upgrade`, alternativ komfortabel über die Weboberfläche unter „Updates“
- Die meisten Updates laufen ohne Reboot durch – betroffen sind meist nur einzelne Dienste, die automatisch neu gestartet werden
- Bewährte Praxis: neue Updates zuerst in einer Testumgebung fahren, erst danach auf Produktiv- bzw. Kundensysteme ausrollen

Wann ist ein Reboot nötig?



- Reboot nur bei zwei Anlässen nötig: einem neuen Kernel oder einem größeren Versionssprung (z. B. Proxmox VE 7 auf 8)
- Grund: ein neuer Kernel wird erst beim nächsten Bootvorgang tatsächlich geladen, bis dahin läuft weiter der alte im Arbeitsspeicher
- Proxmox hält mehrere Kernel-Versionen parallel vorrätig – im GRUB-Menü lässt sich im Problemfall auf die vorherige Version zurückfallen, ohne neu zu installieren
- Real: alle paar Wochen bis Monate ein Kernel-Reboot – damit planbar statt spontan
- Reboot lässt sich fest ins eigene Wartungsfenster legen – kein Überraschungsneustart mitten am Vormittag
- Kein Zwang zu ständigen Neustarts wie man es von manchen Windows-Patchdays kennt

Rolling Reboot ohne VM-Downtime



Ohne Cluster gibt es keine Live-Migration: Ein einzelner Host, der neu startet, nimmt seine VMs zwangsläufig mit in die Downtime. Schon ein einfaches Zwei-Node-Setup mit einem kleinen Quorum-Gerät (QDevice) zahlt sich beim ersten Kernel-Update aus.

- Im Cluster: VMs per Live-Migration auf einen anderen Node verschieben, ganz ohne spürbare Downtime
- Voraussetzung: geteilter oder replizierter Storage sowie ein ausreichend dimensioniertes Migrationsnetz
- Bei Shared Storage wandert nur der Arbeitsspeicher-Zustand, das geht schnell; bei lokalem Storage mit Replikation dauert es etwas länger, ist aber ebenfalls machbar
- Host rebooten, VMs anschließend zurückholen
- Vergleichbar mit vMotion + Wartungsmodus – bei Proxmox ohne Lizenzaufpreis inklusive

Ressourcen erweitern: was live, was mit Shutdown

vDisk vergrößern	vCPU hinzufügen	RAM erhöhen
✓ Live möglich	✓ Live möglich	✓ Live möglich
Voraussetzung Disk-Resize per GUI oder qm resize	Voraussetzung CPU-Hotplug aktiv + Gast unterstützt es	Voraussetzung Memory-Hotplug + passende VM-Konfig
i Gast-FS separat erweitern	✗ sonst: Shutdown nötig	✗ sonst: Shutdown nötig

Ballooning ändert nur das dem Gast aktuell zugeteilte RAM dynamisch innerhalb eines vorher gesetzten Rahmens (Minimum/Maximum) – das ist **kein Ersatz** für ein echtes Erhöhen des Max-RAM.

vDisk vergrößern

Live möglich

Per GUI oder `qm resize` – keine VM-Downtime auf Hypervisor-Ebene. Im Gast müssen Partition und Dateisystem noch erweitert werden (Windows: Datenträgerverwaltung/diskpart; Linux: `growpart` + `resize2fs/xfs_growfs`) – meist ohne Neustart, abhängig von OS und Partitionslayout.

vDisk verkleinern

Nicht live

Nicht live möglich und riskant – in der Regel nur über Neuanlage einer kleineren Disk und Migration der Daten umsetzbar.

Neue vDisk hinzufügen

Live möglich

Hotplug bei VirtIO-SCSI – eine zusätzliche Disk lässt sich im laufenden Betrieb ergänzen.

vCPU hinzufügen

Bedingt live

Nur mit aktiviertem CPU-Hotplug und passender Gast-Unterstützung (Linux meist ja, Windows eingeschränkt). Ohne Hotplug: Gast herunterfahren, ändern, wieder starten.

vCPU reduzieren

Nur mit Shutdown

Reduzieren ist praktisch immer nur im heruntergefahrenen Zustand möglich.

RAM erhöhen

Bedingt live

Nur mit Memory-Hotplug und passender VM-/Gast-Konfiguration. Ohne Hotplug: Shutdown nötig.

RAM-Maximum reduzieren

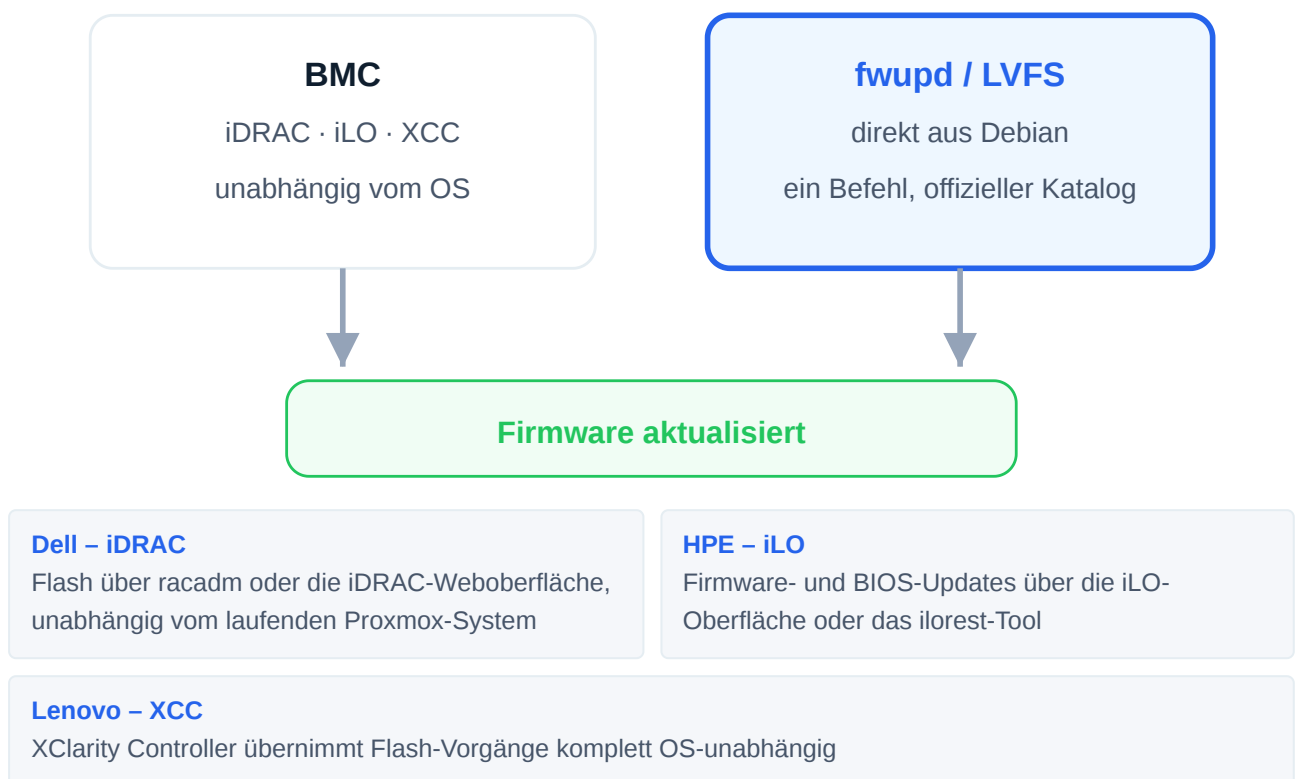
Nur mit Shutdown

Das Reduzieren des Maximums ist in der Regel nur mit Shutdown möglich.

Hotplug-Geräte generell: Disk & NIC via VirtIO-SCSI

- Proxmox unterstützt Hotplug nicht nur für den Umfang einzelner Disks, sondern auch für ganze zusätzliche Geräte: neue virtuelle Disks und Netzwerkkarten (NICs) lassen sich im laufenden Betrieb hinzufügen
- Voraussetzung ist, dass der Gast **VirtIO-Treiber** nutzt (`VirtIO-SCSI` für Disks, `VirtIO` für Netzwerkkarten) – bei IDE/SATA-Emulation oder älteren Gästen ohne VirtIO-Unterstützung ist meist ein Shutdown nötig
- Damit lässt sich z. B. kurzfristig zusätzlicher Storage oder ein weiteres Netzwerksegment anbinden, ohne die laufende VM anzuhalten
- Auch das Entfernen von Hotplug-Geräten funktioniert im laufenden Betrieb, sofern der Gast das Gerät sauber aushängt bzw. deaktiviert

Firmware-Updates



- Alle drei arbeiten über den BMC direkt am Server-Mainboard – Firmware lässt sich so flashen, auch wenn Proxmox selbst gerade gar nicht bootet
- `fwupd / LVFS` direkt aus Debian für viele Standardgeräte – Abfrage per `fwupdmgr get -updates`, Einspielen per `fwupdmgr update`
- Kein herstellerspezifisches Custom-ISO wie bei ESXi nötig (dort binden Dell/HPE eigene Treiberpakete direkt ins Installationsmedium ein)

Herstellertreiber



- Mainline-Linux-Kernel: die meisten Treiber sind schon dabei – Netzwerkkarten (Intel, Broadcom, Mellanox) und RAID-/HBA-Controller (LSI/Broadcom MegaRAID) laufen in aller Regel sofort
- Kein VIB-Modell wie bei ESXi – kein Suchen nach passenden Paketen pro Karte, keine Inkompatibilität nach einem Versions-Upgrade durch fehlende VIBs
- Hardware läuft dadurch oft schon beim ersten Booten, ganz ohne zusätzliche Treiberinstallation
- Bei sehr neuer Hardware hilft bei Bedarf ein optional installierbarer, neuerer Kernel (aktuelleres `pve-kernel`-Paket), falls der Standardkernel ein Bauteil noch nicht kennt

Disk- & RAID-Monitoring

```
$ smartctl -H /dev/sda
SMART overall-health:      PASSED

$ zpool status
pool: rpool
state:  ONLINE
errors: No known data errors

smartd + zpool: früh warnen statt spät reagieren
```

Diese Hersteller-Tools regelmäßig per Cronjob oder Monitoring-Check auslesen – sonst fällt ein degradierter RAID-Verbund erst beim zweiten Plattendefekt auf.

storcli

Broadcom/LSI – aktuelle MegaRAID-Generation

perccli

Dell PERC – Broadcom-Basis mit Dell-Branding

megacli

Ältere MegaRAID-Controller (Legacy)

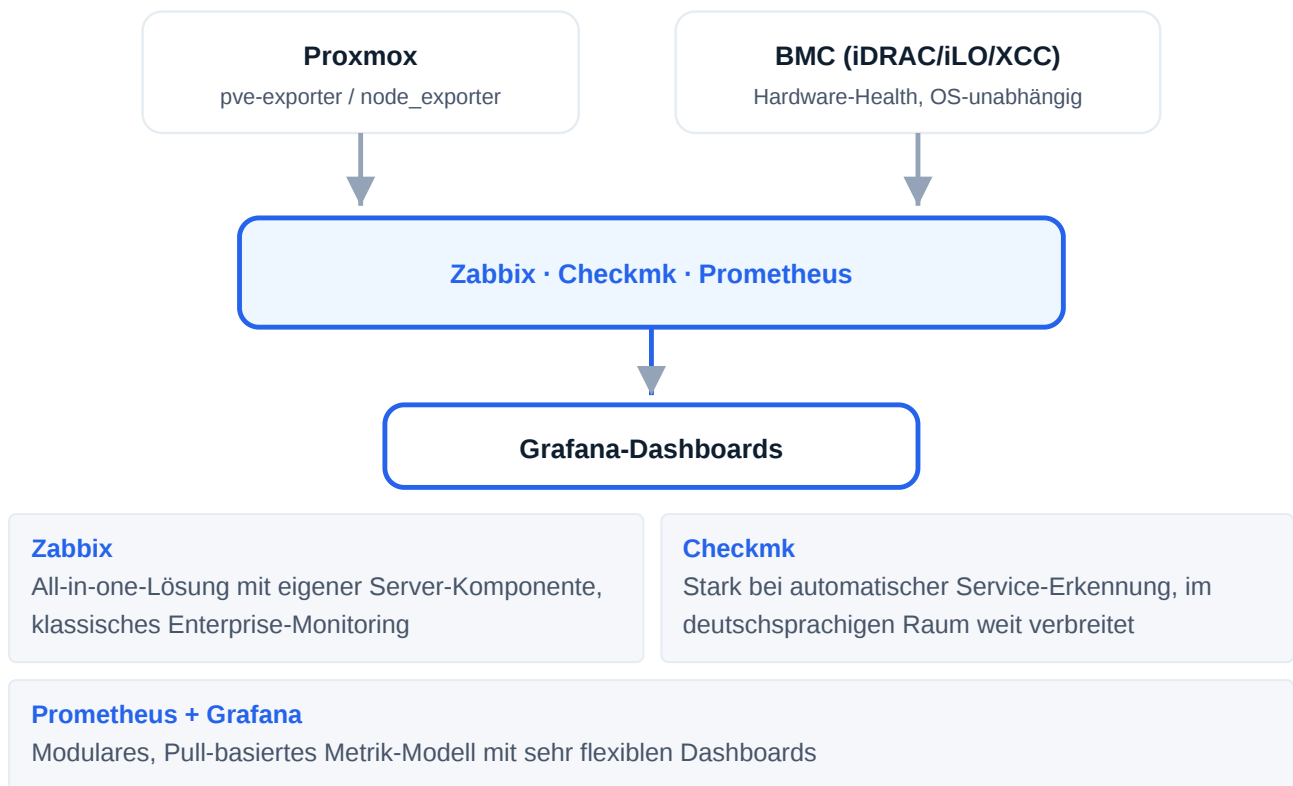
- S.M.A.R.T. über `smartd` ist bereits integriert und meldet eine sterbende Platte, bevor sie ganz ausfällt – Konfiguration in `/etc/smartd.conf`, manuelle Prüfung jederzeit per `smartctl -a /dev/sda`
- Bei ZFS zusätzlich der Pool-Status im Blick behalten (`zpool status`) sowie ein regelmäßiger Scrub (z. B. monatlich), der stille Datenfehler proaktiv aufdeckt

Benachrichtigungen (eingebaut)

Benachrichtigungen (Datacenter)	
EREIGNIS	ZIEL
Backup fehlgeschlagen	E-Mail
Replikation hakt	Gotify
HA-Ereignis (Failover)	E-Mail + Gotify
Seit PVE 8 fest eingebaut, keine Fremdsoftware nötig	

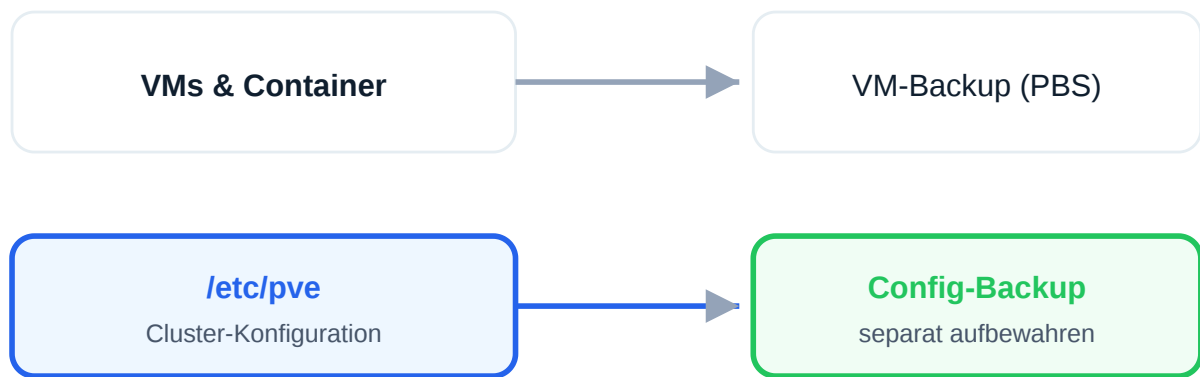
- Eingebautes Notification-System seit Proxmox VE 8, konfigurierbar unter „Rechenzentrum → Benachrichtigungen“
- E-Mail oder Gotify bei Backup-, Replikations- und HA-Ereignissen sowie bei ablaufenden Zertifikaten
- Ziele (wer wird benachrichtigt) und Matcher/Filter (welche Ereignisse) lassen sich getrennt konfigurieren und flexibel kombinieren
- Für den Einstieg oder kleine Umgebungen bereits ausreichend, um sofort zu merken, wenn z. B. das nächtliche Backup nicht durchgelaufen ist

Externes Monitoring



- Proxmox-spezifische Metriken liefert der (Community-) `pve-exporter`, allgemeine System-Metriken wie CPU/RAM/Disk-I/O der `node_exporter`
- Tiefes Hardware-Health (Lüfterdrehzahl, Netzteilstatus, Temperaturen) zusätzlich über den BMC abfragen, z. B. per Redfish-API oder `ipmitool` – Proxmox selbst bildet das nicht so eng ab wie vCenter
- Kombination aus externem Monitoring und BMC-Anbindung ergibt den Rundumblick, den man von vCenter gewohnt ist
- Für Dienstleister besonders wertvoll: ein zentrales Dashboard über alle betreuten Kundenumgebungen hinweg statt Insellösung pro Kunde

Config-Backup & Recovery

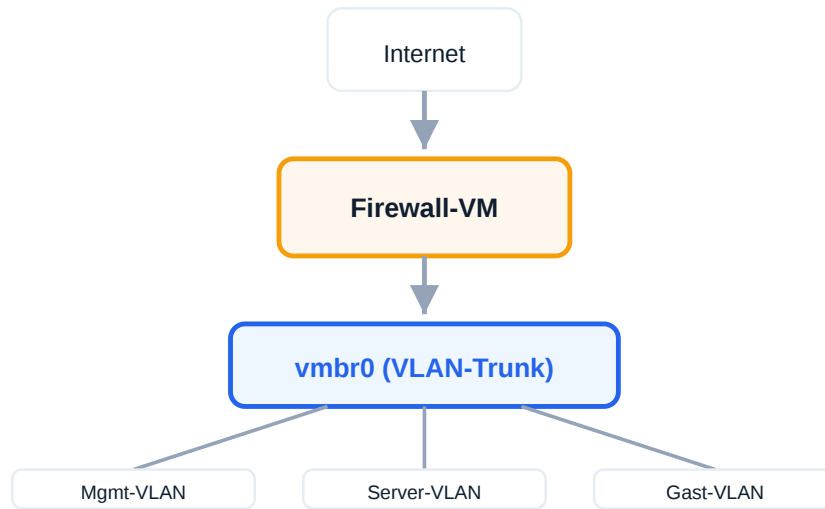


Nur beides zusammen stellt den ganzen Cluster wieder her

- `/etc/pve` sichern – dort liegt die komplette Cluster-Konfiguration (VM-Definitionen, Storage- und Netzwerk-Einstellungen, Benutzer und Berechtigungen)
- `/etc/pve` ist technisch kein normales Dateisystem, sondern die Cluster-Dateisystem-Sicht (`pmxcfs`) auf eine SQLite-Datenbank unter `/var/lib/pve-cluster/config.db` – ein normales `vzdump`-VM-Backup sichert diesen Teil nicht mit
- Regelmäßig separat sichern, z. B. per einfachem `tar`-Archiv von `/etc/pve` oder einer eigenen Backup-Routine für die Konfigurationsdatenbank
- Node- und Cluster-Wiederherstellung vorher durchdenken und idealerweise einmal in einer Testumgebung geübt haben, nicht erst im Ernstfall
- Merkregel: die VM ist das Auto, die Cluster-Konfiguration ist der Fahrzeugschein – beides zusammen ergibt erst den vollständig wiederherstellbaren Betrieb

M6 · Virtualisierte Netzwerkkumgebung

Virtuelle Switches, VLANs, Firewalls als VM – die häufigste Fehlerquelle bei der Migration



Virtuelle Switches

Linux Bridge

Standard, robust
einfache VLAN-Trennung

passt für die meisten KMU

Open vSwitch

erweiterte Funktionen
z. B. komplexes Tunneling

selten nötig im KMU

Linux Bridge

Standardmäßig bei der Installation angelegt (vbr0), robust, für die meisten Setups völlig ausreichend

Open vSwitch

Per apt nachinstallierbar, erweiterte Funktionen wie LACP-Bonding und flexibleres VLAN-Tagging

- Jede physische Netzwerkkarte lässt sich einer Bridge zuordnen, mehrere VMs teilen sich anschließend dieselbe Bridge wie an einem physischen Switch
- Vergleichbar mit dem vSphere Distributed Switch (vDS) bzw. dem Hyper-V-vSwitch, funktional aber ohne separate Lizenzstufe
- Für die meisten KMU-Umgebungen reicht die einfache Linux Bridge völlig aus, Open vSwitch lohnt sich erst bei fortgeschrittenen Netzwerk-Anforderungen

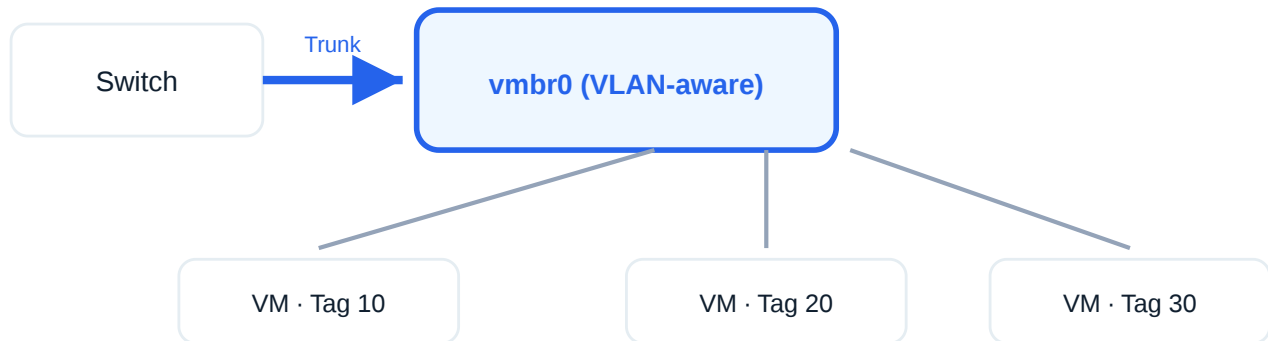
SDN-Feature



Zentral definiert, auf jedem Node automatisch verfügbar

- SDN steht für Software Defined Networking – Netzsegmente werden zentral definiert statt auf jedem Node einzeln von Hand konfiguriert
- **Zonen** bündeln die Netzwerktechnik: **Simple** (isoliertes NAT-Netz), **VLAN** (klassisches Tagging), **QinQ** (doppeltes VLAN-Tagging für Provider-Szenarien) und **VXLAN/EVPN** (Layer-2-Overlay über Layer-3-Routing hinweg, auch über Standortgrenzen hinweg)
- **VNets** sind die eigentlichen virtuellen Netze innerhalb einer Zone, denen VMs zugewiesen werden – vergleichbar mit einer Portgroup
- Konfiguration gilt clusterweit über mehrere Nodes hinweg – einmal definieren, überall im Cluster verfügbar
- Sinnvoll für komplexere oder mandantenfähige Setups, etwa wenn mehrere Kunden strikt getrennte Netze auf derselben Cluster-Hardware erhalten sollen
- Für eine Handvoll VMs auf einem Host ist SDN Überdimensionierung – hier reicht die klassische Bridge-Konfiguration

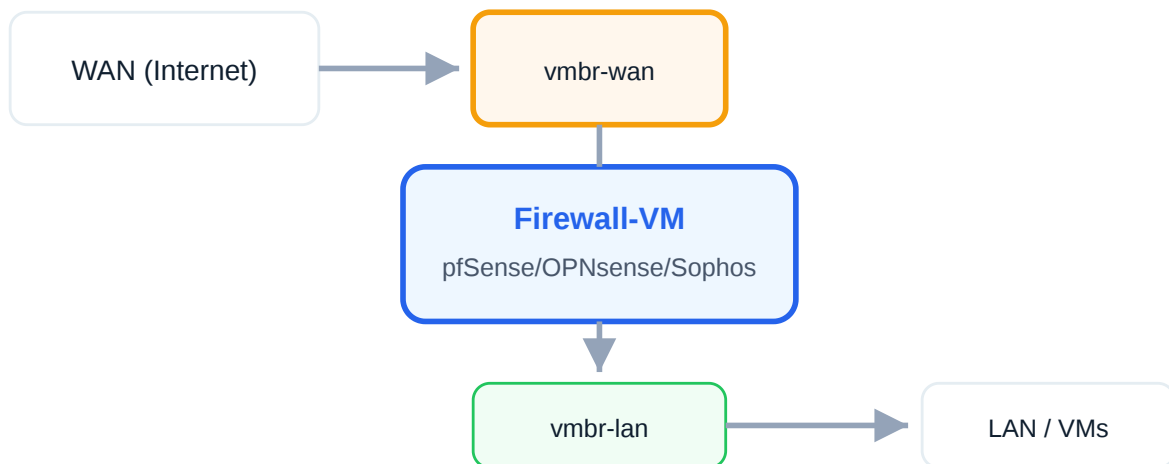
Multi-VLAN-Design



Nach jeder Migration wirklich jedes VLAN einzeln durchtesten, nicht nur eines: Ein vergessenes oder falsch gesetztes Tag versteckt sich erfahrungsgemäß gern im dritten oder vierten VLAN – und fällt dann erst Tage später auf, wenn genau dieses Segment gebraucht wird.

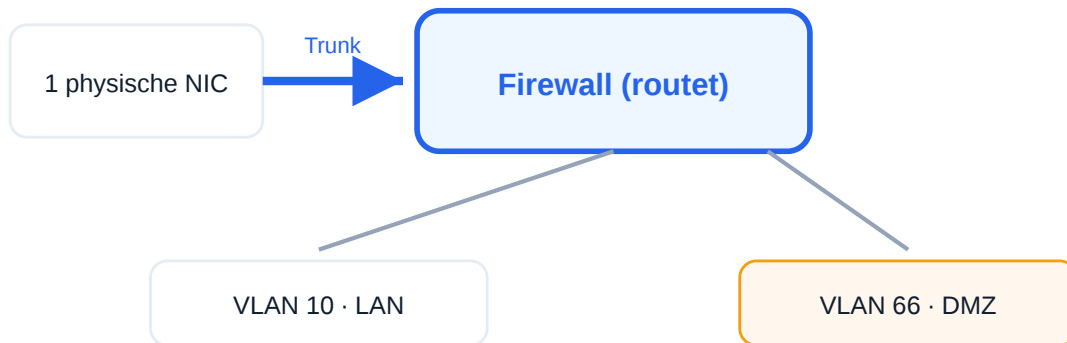
- VLANs sind bei Migrationen die häufigste Fehlerquelle: Stimmt das Tagging irgendwo zwischen physischem Switch, Bridge und VM nicht, ist die VM stumm, obwohl scheinbar alles läuft
- Bridge dazu als **VLAN-aware** markieren – dann kommt der getaggte Trunk vom physischen Switch mit allen VLANs geschlossen bei der Bridge an
- Tagging erfolgt am Trunk-Port des physischen Switches; in Proxmox trägt man je VM-NIC lediglich das gewünschte VLAN-Tag ein, die Bridge übernimmt das Sortieren
- Ein einziges Kabel genügt so für zwanzig oder mehr VLANs gleichzeitig
- Management- und Nutz-VLAN unbedingt trennen – Management und Kundenverkehr im selben VLAN sind gleichzeitig ein Sicherheits- und ein Ausfallrisiko

Virtualisierte Firewalls



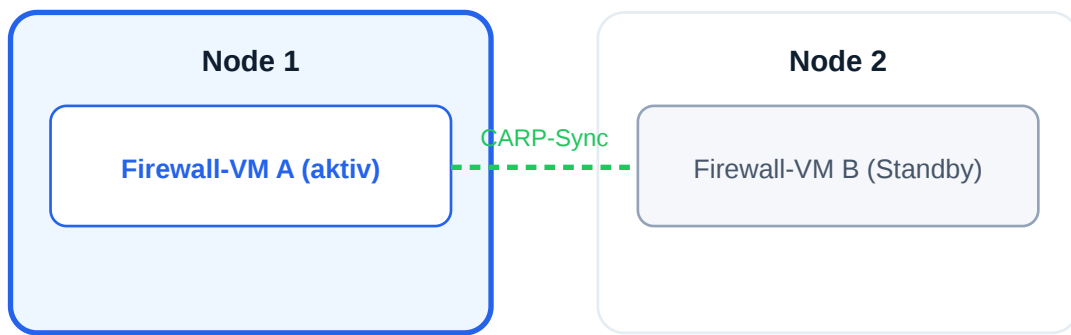
- pfSense, OPNsense oder auch Sophos laufen problemlos als VM auf demselben Host wie die übrigen Workloads – ein etabliertes, weit verbreitetes Muster
- WAN und LAN als zwei getrennte Bridges führen, damit die Firewall wirklich zwischen den Netzen steht statt nur virtuell danebenzustehen
- Die Firewall-VM erhält entsprechend zwei virtuelle NICs – eine Richtung Internet/Uplink, eine Richtung internes Netz
- VirtIO-NICs für Durchsatz und geringen Overhead einplanen, dazu ausreichend vCPUs – viele Firewall-Distributionen verarbeiten Pakete eher seriell pro Kern, eine unterdimensionierte Firewall-VM wird schnell selbst zum Flaschenhals
- Für das KMU-Referenz-Szenario dieses Lernbereichs (1–2 Nodes, 4–6 VMs, siehe Modul 0) reicht eine einzelne Firewall-VM mit moderater Ausstattung (2–4 vCPUs) in aller Regel aus – zusätzlich spart die Netzwerksegmentierung über VLANs, die eine solche Firewall-VM durchsetzt, direkt auf die Risikomanagement-Maßnahmen ein, die NIS2 von vielen Unternehmen verlangt (siehe Modul 4)

Design: Firewall-on-a-Stick



- Eine einzige physische NIC als VLAN-Trunk genügt – alle benötigten VLANs laufen getaggt über diesen einen Uplink
- Die Firewall-VM selbst übernimmt das Routing zwischen den VLANs, inklusive Filterregeln pro Segment
- DMZ-Segmentierung ist damit inklusive, ganz ohne zusätzliche Verkabelung oder weitere physische NICs
- Kehrseite: der gesamte Verkehr aller VLANs läuft über diesen einen Trunk und durch diese eine Firewall-Instanz – für die meisten KMU-Umgebungen ausreichend dimensioniert, bei hohem Durchsatzbedarf aber bewusst einplanen
- Fällt dieser eine Uplink oder die Firewall-VM aus, steht das komplette geroutete Netz – ein Grund, warum bei höheren Verfügbarkeitsanforderungen die doppelt ausgelegte HA-Firewall aus dem nächsten Block sinnvoll wird

HA-Firewall



Fällt Node 1 aus, übernimmt VM B in Sekunden

- Zwei Firewall-VMs auf zwei unterschiedlichen Nodes betreiben – auf demselben Node hätte ein Node-Ausfall beide gleichzeitig lahmgelegt, das wäre keine echte Hochverfügbarkeit, nur doppelter Aufwand
- Synchronisation über **CARP** (Common Address Redundancy Protocol) für die virtuelle IP sowie Konfigurations- und State-Sync (bei pfSense/OPNsense z. B. via `pf sync`) zwischen den beiden Instanzen
- Fällt ein Node aus, übernimmt die zweite Firewall binnen Sekunden, ohne dass bestehende Verbindungen abreißen
- Wichtig: nicht nur die Konfiguration, auch der Verbindungsstatus (State Table) sollte repliziert werden, sonst müssen aktive Sessions nach dem Failover neu aufgebaut werden

Fallstricke

⚠ Fallstricke bei virtualisierten Firewalls

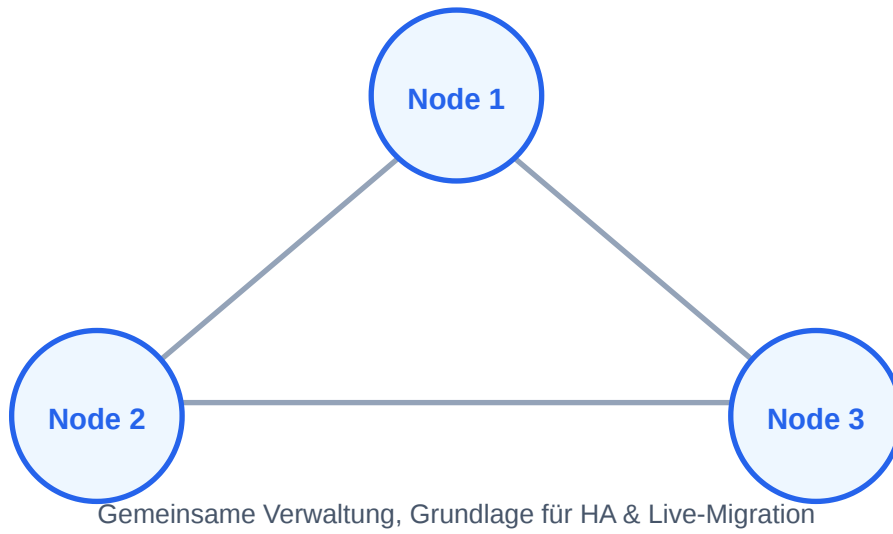
- VirtIO vs. NIC-Passthrough: Flexibilität vs. Durchsatz abwägen
- Hardware-Offloading kann Durchsatz mysteriös einbrechen lassen
- MAC-Spoofing/Promiscuous-Mode gezielt erlauben, wo nötig
- Netzwerkschleifen (Loops) vermeiden – legen ganze Segmente lahm

Netzwerkschleifen (Loops) unbedingt vermeiden: Werden versehentlich zwei Uplinks ohne Spanning Tree zusammen gebridged, kann ein einziger Loop das komplette Netzwerksegment lahmlegen – Broadcast-Sturm statt Ausfallsicherheit. Physische Switches mit aktiviertem STP/RSTP fangen solche Fehlkonfigurationen zusätzlich ab.

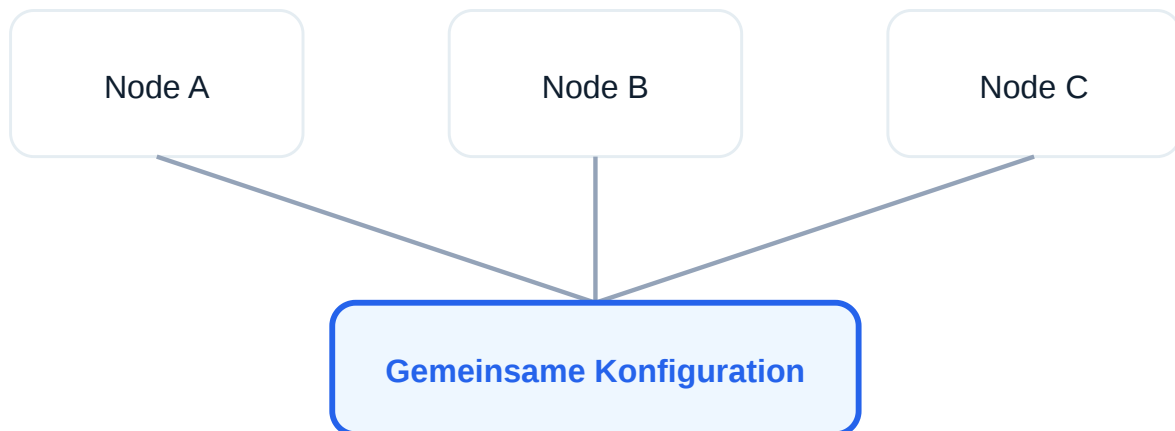
- VirtIO meist die beste Wahl für Netzwerkkarten – schnell, stabil, ohne Sonderkonfiguration; NIC-Passthrough nur in Sonderfällen (z. B. sehr hohe Durchsatzanforderungen), kostet aber Flexibilität wie Live-Migration
- Hardware-Offloading (TSO, GSO, GRO, Checksum-Offload) kann in virtualisierten Firewalls Durchsatz-Probleme verursachen – bei mysteriösen Performance-Einbrüchen im Zweifel einzeln per `ethtool -K` abschalten und testen
- MAC-Spoofing und Promiscuous-Mode gezielt erlauben, wenn die Firewall Pakete für andere MAC-Adressen verarbeiten muss (Netzwerkoptionen der jeweiligen VM-NIC) – ohne diese Freigabe verwirft die Bridge solche Pakete stillschweigend

M7 · Cluster & Quorum

Mehrere Nodes zu einem Verbund – Grundlage für HA und Live-Migration



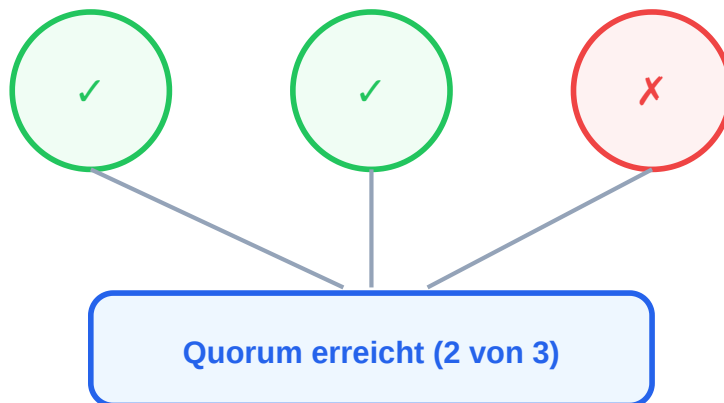
Was ist ein Cluster?



Synchronisierte Uhrzeit auf allen Nodes ist Pflicht, z. B. per `chrony` oder `systemd-timesyncd` – schon eine Abweichung von wenigen Sekunden kann bei Corosync und dem Cluster-Dateisystem zu unnötigen Fehlalarmen und Konfigurationskonflikten führen.

- Mehrere Nodes, eine gemeinsame Verwaltung über eine einzige Web-GUI – egal, an welchem Node man sich anmeldet, man sieht immer den ganzen Cluster
- Gemeinsame Konfiguration liegt unter `/etc/pve` – kein normales Verzeichnis, sondern die **Proxmox Cluster File System (pmxcfs)**, eine Fuse-Datenbank, die über Corosync in Echtzeit auf alle Nodes repliziert wird
- Beitritt per `pvecm add <ip-des-ersten-nodes>` auf der Kommandozeile oder per Klick im Web-Interface – Proxmox tauscht dabei automatisch die nötigen SSH-Schlüssel zwischen den Nodes aus
- Voraussetzung für Live-Migration und Hochverfügbarkeit – ohne Cluster-Verbund bleibt jeder Node eine Insel für sich, mit eigener Konfiguration und eigenem VM-Bestand
- Empfohlene Obergrenze liegt bei etwa 32 Nodes je Cluster; mehr ist technisch möglich, verlangt dann aber ein sauber geplantes, latenzarmes Corosync-Netz
- Diese clusterweite Sicht gilt auch für den Konsolenzugriff: Web-GUI und API kennen alle Nodes und proxyen eine noVNC-/SPICE-Verbindung automatisch zu dem Node, auf dem die jeweilige VM tatsächlich läuft – ein Login an einem beliebigen Node genügt für den Zugriff auf alle VMs im Cluster
- Nach einer Live-Migration ändert sich am Zugriffsweg nichts: die Konsole „folgt“ der VM automatisch zum neuen Node, weil jeder Node denselben, über `pmxcfs` synchronisierten Cluster-Zustand kennt – Details und Voraussetzungen dazu in [Modul 1 – Grundlagen & Architektur](#)

Corosync & Quorum

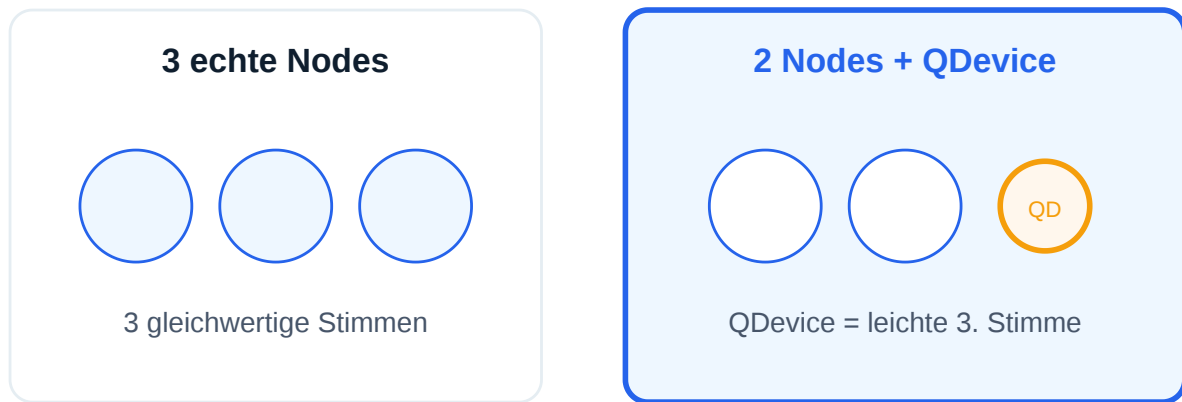


Corosync trägt die Abstimmung zwischen allen Nodes aus

Mit `pvecm status` sieht man auf einen Blick, wie viele Stimmen aktuell vorhanden sind, wie viele für Quorum nötig sind und ob der Cluster gerade quorat ist – der erste Befehl bei jedem Cluster-Problem.

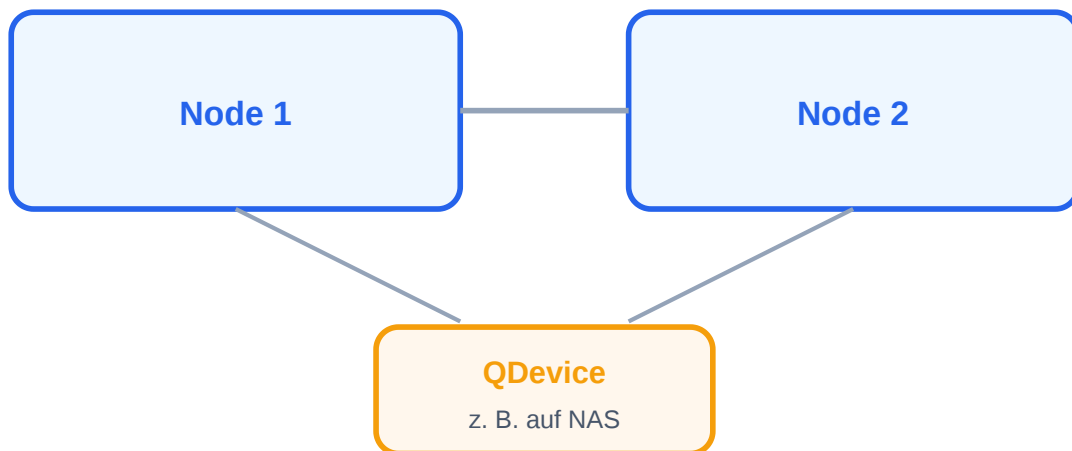
- Corosync ist der Herzschlag-Dienst: über das **Totem-Protokoll** tauschen alle Nodes ständig Status-Pakete aus, standardmäßig über den modernen `knet`-Transport (Nachfolger von klassischem UDP-Multicast)
- Konfiguriert wird das in `/etc/pve/corosync.conf` – Werte wie `token` (Timeout, bis ein Node als tot gilt, Standard 1000 ms) und `expected_votes` stehen genau hier
- Quorum: die Mehrheit der Stimmen muss sich einig sein, bevor VMs gestartet, gestoppt oder verschoben werden – jeder Node bringt in der Standardkonfiguration eine Stimme mit
- Zuständig dafür ist **votequorum**, ein Bestandteil von Corosync – Status jederzeit einsehbar per `pvecm status`
- Verhindert widersprüchliche Entscheidungen bei Netzproblemen: ohne Mehrheit verweigert der Cluster riskante Aktionen, statt zu raten und im Zweifel falsch zu handeln

Wie viele Nodes?



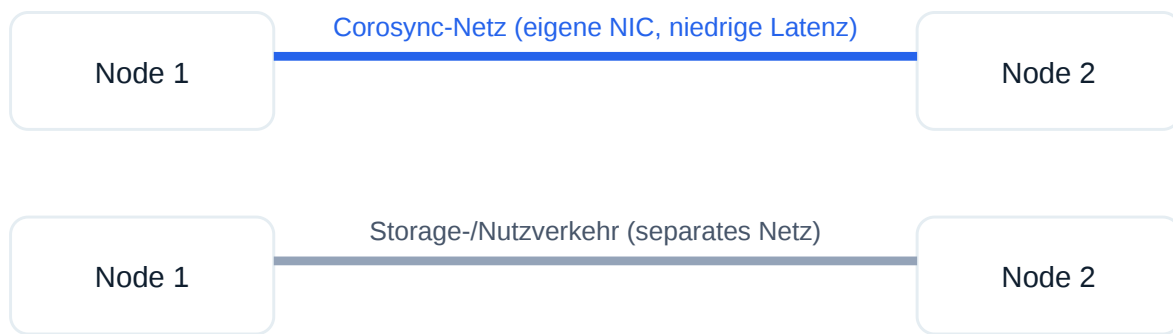
- Mindestens drei Nodes für ein sicheres Quorum – bei drei Stimmen sind zwei für die Mehrheit nötig ($\text{floor}(3/2)+1 = 2$), der Cluster verkraftet also genau einen Node-Ausfall
- Fünf Nodes verkraften zwei gleichzeitige Ausfälle, sieben Nodes drei – die Ausfalltoleranz wächst mit jeder zusätzlichen Zweiergruppe an Nodes
- Alternativ zwei Nodes plus **QDevice** als dritte, „leichte“ Stimme: `corosync-qnetd` läuft auf einem externen Gerät, `corosync-qdevice` auf beiden Cluster-Nodes
- Ungerade Stimmenzahl wird bevorzugt – vermeidet Patt-Situationen wie 2 zu 2 bei vier Nodes, bei denen keine Seite eine Mehrheit erreicht
- Zwei Nodes ganz ohne dritte Stimme sind für Testbetrieb in Ordnung, für Hochverfügbarkeit aber ungeeignet: fällt einer aus, steht es 1:1, und der verbleibende Node bekommt kein Quorum
- Für das KMU-Referenz-Szenario dieses Lernbereichs (1–2 Nodes, siehe Modul 0) ist zwei Nodes plus QDevice damit fast immer die richtige Zielgröße – drei echte Nodes lohnen sich meist erst, wenn ohnehin mehr Rechenleistung gebraucht wird

Best Practice 2-Node-Setup



- QDevice (z. B. auf NAS, Mini-PC oder einem Raspberry-Pi-Klasse-Gerät) ist zwingend – ohne dritte Stimme bleibt der 2-Node-Cluster für HA unbrauchbar
- Einrichtung in zwei Schritten: `apt install corosync-qnetd` auf dem externen Gerät, danach `pvecm qdevice setup <ip-des-qnetd>` auf einem der beiden Cluster-Nodes – Proxmox verteilt die Konfiguration automatisch an beide Nodes
- Der QDevice selbst hostet keine VMs und trägt keine Nutzlast, er beantwortet im Zweifel nur die Quorum-Abstimmung – entsprechend genügt sehr bescheidene Hardware
- Wichtig für die Platzierung: möglichst eigene Stromversorgung und eigener Netzwerk-Switch – sonst reißt ein einzelner Stromausfall Node und QDevice gleichzeitig weg, und die dritte Stimme nützt nichts
- Zwei Nodes allein ergeben kein sicheres Quorum: bei Ausfall eines Node steht es 1:1, keine Mehrheit, HA-Ressourcen bleiben eingefroren statt automatisch umzuziehen
- Damit wird das Setup erst richtig HA-fähig für KMU – bei überschaubaren Zusatzkosten für ein einziges kleines QDevice-Gerät

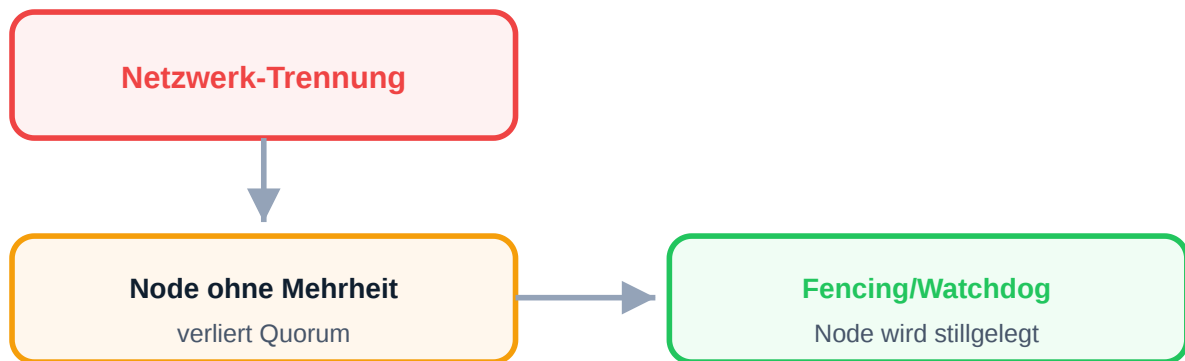
Dediziertes Corosync-Netz



Trennung verhindert Cluster-Zittern unter Last

- Eigene NIC (oder zumindest ein eigenes VLAN) für Corosync, mit niedriger und vor allem konstanter Latenz – Corosync reagiert empfindlicher auf Latenz-Spitzen als auf die absolute Bandbreite
- Getrennt vom Storage- und Nutzverkehr, idealerweise auch vom Migrations- und Backup-Netz – ein sättigender Backup-Job auf derselben Leitung kann Herzschläge spürbar verzögern
- Corosync unterstützt redundante Links über mehrere Ringe (`knet`) – ein zweites, physisch getrenntes Netz für Corosync selbst erhöht die Ausfallsicherheit der Cluster-Kommunikation nochmal zusätzlich
- Verhindert Cluster-Zittern unter Last: wird das `token -Timeout` (Standard 1000 ms) wiederholt überschritten, stuft der Cluster einen eigentlich gesunden, nur überlasteten Node fälschlich als tot ein
- Praxiswert: deutlich unter 2 ms Round-Trip-Zeit gilt als komfortabel – über WAN-Strecken oder stark ausgelastete Switches wird Corosync schnell nervös und meldet Fehlalarme

Split-Brain & Fencing

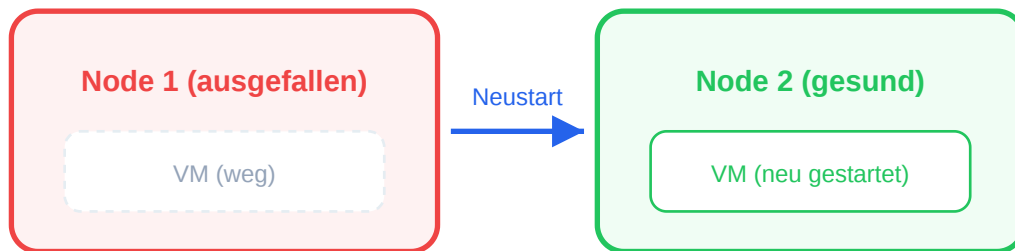


Lieber ein Node aus, als zwei im Datenchaos

- Fencing/Watchdog verhindert doppelten VM-Start bei Split-Brain – Proxmox setzt dabei standardmäßig auf einen Software-Watchdog (Kernel-Modul `softdog`, Gerät `/dev/watchdog`) statt auf klassisches externes IPMI-Fencing
- Das Prinzip ist „Selbst-Fencing“: verliert ein Node das Quorum, füttert der `watchdog-mux`-Dienst den Watchdog-Timer nicht mehr weiter – nach spätestens rund 60 Sekunden löst die Hardware einen harten Reset dieses Nodes aus, ganz ohne Eingriff eines anderen Nodes
- Der spezielle `two_node`-Modus (kombiniert mit `wait_for_all` in `corosync.conf`) ist nur ohne echte HA vertretbar – er erzwingt Quorum bereits bei zwei Stimmen und wäre bei aktivem HA-Manager ein Split-Brain-Risiko
- Sicherheit vor Datenchaos hat Vorrang vor Weiterlaufen um jeden Preis: lieber ein Node kontrolliert per Watchdog-Reset aus dem Rennen nehmen, als riskieren, dass zwei Nodes gleichzeitig auf dieselbe VM-Disk schreiben
- Ohne funktionierenden Watchdog im Server (BMC/IPMI-Reset oder Software-Watchdog) startet der HA-Stack aus Sicherheitsgründen erst gar nicht – Fencing ist keine Option, sondern Voraussetzung

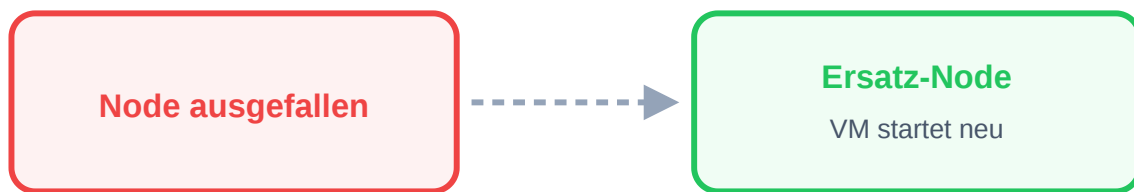
M8 · Hochverfügbarkeit (HA)

Ausfallzeit auf Sekunden senken – was realistisch ist, und was nicht



Automatisch, ohne Handeingriff – Ausfallzeit auf Sekunden senken

Was HA leistet

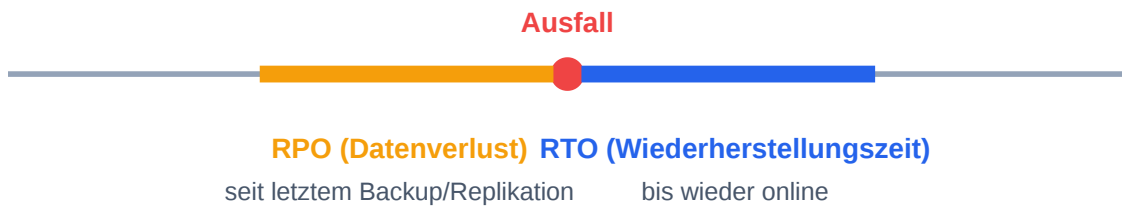


Kein unterbrechungsfreier Betrieb – aber automatisch und schnell

Nur VMs und Container, die explizit als HA-Ressource eingetragen sind, werden so behandelt – Verwaltung über `ha-manager` bzw. die Datei `/etc/pve/ha/resources.cfg`, nicht automatisch für jede VM im Cluster.

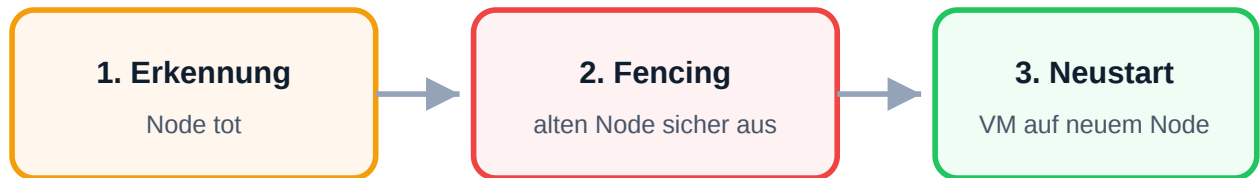
- Node fällt aus → betroffene VM startet automatisch auf einem anderen, gesunden Node neu – auch nachts oder am Wochenende, ohne dass jemand eingreifen muss
- Zwei Dienste teilen sich die Arbeit: der **CRM** (Cluster Resource Manager, aktiv auf dem aktuellen Master) entscheidet, was passieren soll, der **LRM** (Local Resource Manager) auf jedem Node setzt es lokal um
- Kein unterbrechungsfreier Betrieb per se – die VM startet neu, laufende Verbindungen wie RDP-Sitzungen oder offene Datenbank-Transaktionen gehen dabei verloren
- HA-Gruppen (`/etc/pve/ha/groups.cfg`) legen fest, auf welchen Nodes eine Ressource bevorzugt läuft – nützlich, wenn VMs aus Lizenz- oder Performance-Gründen an bestimmte Hardware gebunden bleiben sollen

RTO vs. RPO



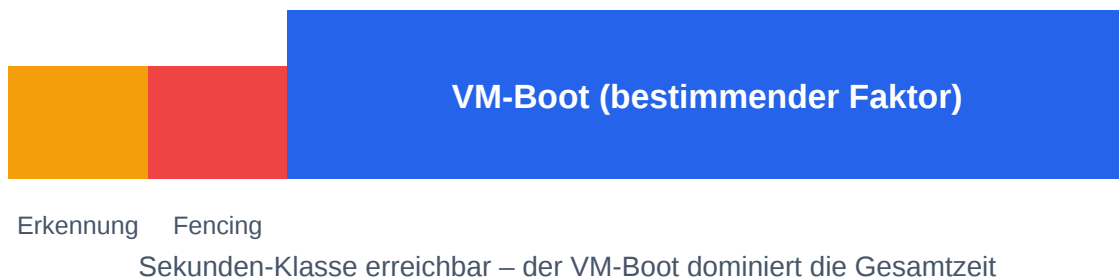
- **RTO** (Recovery Time Objective): wie schnell ist der Betrieb wieder online – bildlich, wie lange das Geschäft geschlossen bleibt
- **RPO** (Recovery Point Objective): wie viele Daten gehen im schlimmsten Fall verloren – bildlich, wie viele der letzten Buchungen im Brandfall verloren gehen
- Zwei getrennte Ziele, die unterschiedliche Stellschrauben betreffen: RTO hängt an Erkennung, Fencing und VM-Boot, RPO hängt fast ausschließlich am gewählten Storage
- Beide Werte sollten vorab mit dem Kunden konkret beziffert werden, z. B. „RTO unter 2 Minuten“, „RPO unter 1 Minute“ – ohne Zahl bleibt „hochverfügbar“ eine leere Behauptung
- Faustregel: ein kleineres RPO kostet meist mehr Storage-Aufwand (Ceph statt Replikation), ein kleineres RTO eher schnellere Hardware und schlankere VM-Images

Ablauf eines Failovers



- Erkennung: der Cluster stellt über Corosync/Quorum fest, dass ein Node tot ist – nicht beim ersten verpassten Herzschlag, sondern erst nach Ablauf des konfigurierten Token-Timeouts, um Fehlalarme durch kurze Hänger zu vermeiden
- Fencing: der alte Node wird sicher stillgelegt, in der Praxis meist über den Software-Watchdog (siehe Modul 7) – dieser Schritt dauert bei Proxmox typischerweise bis zu rund einer Minute, weil der Watchdog-Timeout bewusst konservativ eingestellt ist
- Erst wenn das Fencing bestätigt ist, gibt der HA-Manager die Ressource für einen Neustart frei – eine bewusste Sicherheitsbremse, lieber kurz warten als riskieren, dass die VM doppelt läuft
- Neustart der VM auf einem gesunden Node, ausgewählt nach den in den HA-Gruppen hinterlegten Präferenzen und der aktuellen Auslastung der verbleibenden Nodes
- Alle drei Schritte laufen automatisch über `ha-manager`, nachvollziehbar per `ha-manager status` oder in der Web-GUI unter „HA“

„Sekunden statt Minuten" – ehrlich



- RTO = Erkennung + Fencing + VM-Boot – drei Phasen, die sich addieren, nicht überlappen
- Sekunden-Klasse für Erkennung und Fencing erreichbar, in Summe aber oft schon 30 bis 90 Sekunden allein für diese beiden ersten Schritte, je nach Token- und Watchdog-Timeout
- Der VM-Boot bleibt der bestimmende Faktor für die Gesamtzeit: ein schlankes Linux mit wenig Diensten bootet in 10–20 Sekunden, ein „fettes" Windows-Server-System mit vielen Diensten kann durchaus 1–2 Minuten brauchen
- Konsequenz für die Praxis: schlanke, zweckgebundene VM-Images verbessern die reale HA-Zeit oft mehr als jede Feinjustierung am Cluster selbst
- „Sekunden statt Minuten" bezieht sich ehrlicherweise auf die Cluster-Reaktion, nicht zwangsläufig auf die gesamte gefühlte Ausfallzeit bis zur wieder erreichbaren Anwendung

Voraussetzung: Storage

Shared Storage (Ceph)

Daten liegen auf allen Nodes

RPO = 0

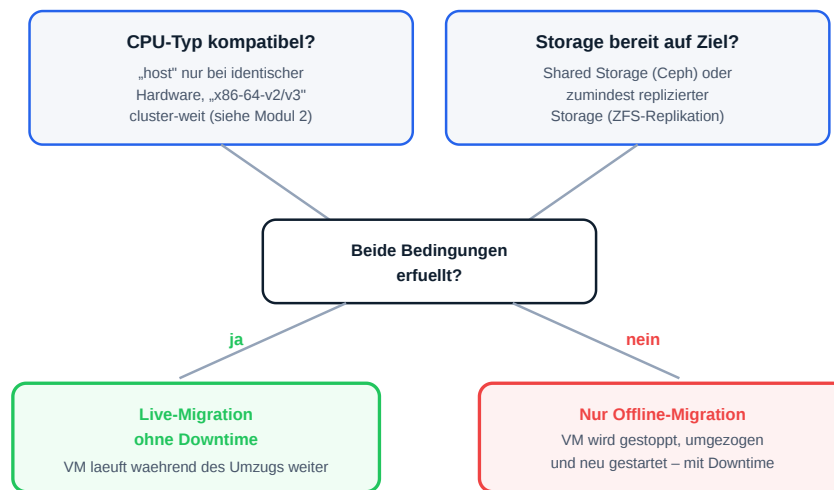
Lokale Replikation

Kopie im Intervall (z. B. 1 Min.)

RPO > 0

- Shared Storage (Ceph) → RPO 0: die Daten liegen durch die verteilte Speicherung ohnehin auf mehreren Nodes gleichzeitig, ein Node-Ausfall ändert daran nichts
- Lokale Replikation → RPO größer 0, abhängig vom Intervall – Proxmox repliziert per `pvesr` standardmäßig alle 15 Minuten, das Intervall lässt sich aber bis auf 1 Minute verkürzen
- Rechenbeispiel: bei 1-Minuten-Intervall gehen im ungünstigsten Fall die Änderungen der letzten bis zu 60 Sekunden vor dem Ausfall verloren – bei 15 Minuten entsprechend bis zu 15 Minuten
- Kürzere Replikationsintervalle kosten mehr I/O- und Netzwerk-Last auf dem Ziel-Node – das Intervall ist ein Kompromiss zwischen RPO und Systemlast, nicht beliebig auf null drehbar
- Der Storage entscheidet über den maximalen Datenverlust – RTO und RPO hängen an komplett unterschiedlichen Bauteilen der Architektur
- „Shared Storage“ heißt in der Proxmox-Praxis meist HCI mit **Ceph**: die Nodes bilden über ihre lokalen Platten gemeinsam einen verteilten Storage-Cluster, es ist kein separates externes SAN nötig – der eigentliche vSAN-Ersatz für HA. Details dazu in Modul 3 („HCI: vSAN-Ersatz mit Ceph“)

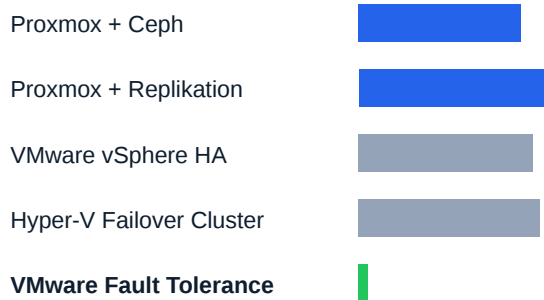
Voraussetzungen für Live-Migration



Dieselben zwei Bauteile – CPU-Typ-Kompatibilität und Storage-Verfügbarkeit auf dem Ziel-Node – entscheiden also sowohl über unterbrechungsfreie Live-Migration als auch über RPO im HA-Failover-Fall. Wer Cluster und Storage von Anfang an darauf auslegt, gewinnt beides gleichzeitig.

- Live-Migration ist der geplante, manuelle Umzug einer laufenden VM auf einen anderen Node – zu unterscheiden vom automatischen HA-Failover nach einem Ausfall, hat aber ähnliche technische Voraussetzungen
- CPU-Typ muss zwischen den Nodes kompatibel sein: `host` erlaubt Live-Migration nur zwischen identischer Hardware, `x86-64-v2` / `x86-64-v3` auch zwischen unterschiedlichen CPU-Generationen (siehe Modul 2, „CPU-Typ in der VM“)
- Shared Storage (z. B. **Ceph**) oder zumindest replizierter Storage (ZFS-Replikation per `pvesr`) nötig, damit die VM-Disk auf dem Ziel-Node bereits vorliegt bzw. innerhalb des Replikationsintervalls schnell verfügbar gemacht werden kann
- Fehlt eine dieser Voraussetzungen – inkompatibler CPU-Typ oder Storage nur lokal ohne Replikation – ist lediglich eine Offline-Migration möglich: die VM wird gestoppt, die Disk kopiert und die VM auf dem Ziel-Node neu gestartet, mit spürbarer Downtime

Ausfall-Zeitvergleich (Node-Ausfall)



alle Balken: Sekunden bis wenige Minuten (VM-Boot-abhängig)

nur Fault Tolerance: praktisch 0 Sekunden, dafür teuer & limitiert

Proxmox + Ceph

Erkennung + VM-Neustart, RPO 0 – vergleichbare Ausfallzeit wie die kommerziellen Lösungen, ohne Lizenzkosten

Proxmox + Replikation

Gleiche Neustart-Mechanik, kleines RPO je nach Intervall – günstigste Variante ab zwei Nodes

VMware vSphere HA

Gleiches Grundprinzip: Erkennung, dann VM-Neustart auf einem anderen Host – ähnliche Zeitklasse wie Proxmox-HA

VMware Fault Tolerance

Sonderfall: laufende Schatten-VM, praktisch 0 Sekunden Unterbrechung – aber teuer und mit harten Größenlimits je VM

Hyper-V Failover Cluster

Ebenfalls Erkennung + VM-Neustart, technisch nah an Proxmox-HA – zusätzliche Lizenzkosten für Windows Server fallen an

Auffällig: fast alle etablierten Lösungen lösen HA nach demselben Muster (Erkennung → Fencing/Isolation → Neustart). Der große Zeitunterschied entsteht erst bei VMware Fault Tolerance, das als einzige Technik kein Neustart-, sondern ein Live-Spiegel-Modell fährt.

Die 0-Sekunden-Wahrheit

VMware Fault Tolerance

Schatten-VM läuft live mit

**0 Sekunden,
aber teuer & limitiert**

Alle HA-Restart-Lösungen

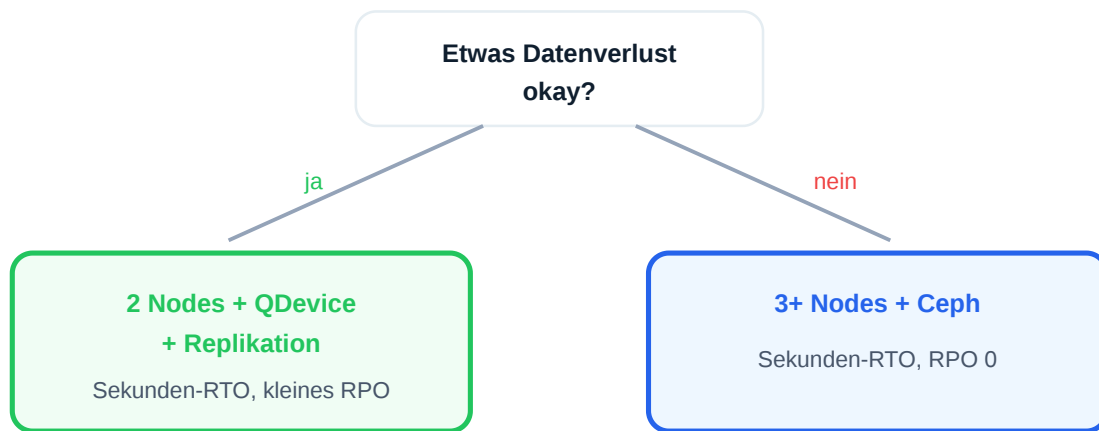
Proxmox, VMware HA, Hyper-V

**VM-Boot nötig,
dafür kostenlos/günstig**

Wichtiger Denkfehler in der Praxis: Hypervisor-HA fängt Hardware-Ausfälle ab, sie ersetzt keine fehlende Redundanz in der Anwendung selbst. Wer Nullausfall verspricht, ohne die Anwendung selbst redundant auszulegen, verspricht zu viel.

- Echte 0 Sekunden nur mit VMware Fault Tolerance – mit harten Limits: begrenzte vCPU-Anzahl je geschützter VM, zusätzlicher Ressourcenbedarf durch die permanent mitlaufende Schatten-VM und spürbarer Lizenzaufpreis
- Alle „HA-Restart“-Lösungen (auch Proxmox, VMware vSphere HA, Hyper-V Failover Cluster) brauchen einen VM-Boot – der Unterschied zwischen den Anbietern liegt eher in Details als im Grundprinzip
- Für echte Unterbrechungsfreiheit: App-HA innerhalb der VM bzw. auf Anwendungsebene einplanen, z. B. ein Datenbank-Cluster mit mehreren aktiven Knoten (etwa Galera oder Patroni) oder zwei Webserver hinter einem Loadbalancer

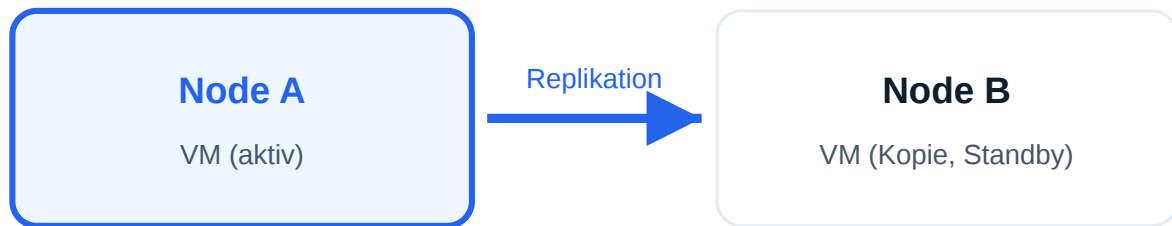
Empfehlung für KMU



- 2 Nodes + QDevice + Replikation: Sekunden-Klasse-RTO, kleines und kalkulierbares RPO (abhängig vom Replikationsintervall) – für die meisten kleinen Firmen der pragmatische Einstieg
- 3+ Nodes + Ceph: ebenfalls Sekunden-Klasse-RTO, dazu RPO 0, weil die Daten ohnehin verteilt liegen – mehr Hardware- und Netzaufwand, dafür echte Nullverlust-Sicherheit
- Leitfrage für die Beratung: „Was passiert, wenn die letzte Minute an Daten fehlt?“ – beim Handwerksbetrieb meist ein Schulterzucken, bei einer Arztpraxis mit laufender Abrechnung oder einem Online-Shop mit Bestellungen ein echtes Problem
- Sizing richtet sich nach dem tatsächlichen Schutzbedarf, nicht nach Technik-Vorliebe: Ceph beim Handwerksbetrieb ist teure Über-Ausstattung, Replikation bei der Arztpraxis ein vermeidbares Risiko
- Die Entscheidung sollte vor dem Hardware-Kauf fallen, nicht danach – Node-Anzahl und Storage-Netz für Ceph sind deutlich aufwendiger nachzurüsten als von Anfang an einzuplanen
- Für das KMU-Referenz-Szenario dieses Lernbereichs (1–2 Nodes, 4–6 VMs, siehe Modul 0) heißt das konkret: 2 Nodes + QDevice + Replikation deckt die Redundanz-Anforderungen der allermeisten Betriebe dieser Größe zuverlässig ab, ohne den Aufwand eines Ceph-Aufbaus

M9 · Replikation

VMs zwischen Nodes synchron halten – Grundlage für HA ohne Shared Storage



Grundlage für HA ohne Shared Storage

Was ist Replikation?



Gesteuert über `pvesr`, technisch ZFS send/receive

Replikation ersetzt kein Backup: Sie hält nur die beteiligten Nodes synchron. Ein versehentlich gelöscht File oder eine fehlerhafte Anwendungsänderung wird ebenso zuverlässig repliziert wie ein korrekter Datensatz.

- Basiert auf ZFS send/receive, automatisiert und geplant über den Dienst `pvesr` (Proxmox VE Storage Replication) – kein manuelles Hantieren mit `zfs send/zfs receive` nötig
- Kopiert VM- und LXC-Container-Daten in festen Intervallen von einem Node auf einen zweiten
- Nur Deltas werden übertragen, nicht jedes Mal die komplette VM – wurden seit dem letzten Lauf z. B. nur 800 MB geändert, wandern auch nur diese 800 MB, nicht die komplette 100-GB-Disk erneut
- Voraussetzung: beide Nodes brauchen ZFS-Storage mit identischem Storage-Namen im Cluster – Replikation funktioniert ausschließlich auf ZFS, nicht auf LVM-thin, ext4 oder NFS
- Jeder Lauf legt intern ein Snapshot-Paar an, aus dessen Differenz das Delta berechnet wird; ältere Snapshots räumt `pvesr` automatisch auf

Einrichtung in der Praxis

Migration einer VM zwischen ihren beiden Replikationspartnern nutzt den bereits synchronen Stand: Es muss nur noch das letzte Delta übertragen werden – spürbar schneller als eine „kalte“ Migration auf einen fremden Node ohne bestehende Replikationsbeziehung.

GUI

Datacenter → Replication → Add: VM auswählen, Ziel-Node, Zeitplan und optional ein Bandbreitenlimit festlegen

CLI

`pvesr create-local-job` – dieselbe Konfiguration per Kommandozeile, praktisch für Skripte und Automatisierung

- Beispiel: `pvesr create-local-job 100-0 pve2 --schedule '*/15' --rate 50` repliziert VM 100 alle 15 Minuten auf Node pve2, gedeckelt auf 50 MB/s
- Status prüfen mit `pvesr status` – zeigt je Job letzten Lauf, nächsten Lauf, Dauer und Fehlerzähler
- Konfiguration liegt clusterweit synchron in `/etc/pve/replication.cfg` – wie alle Dateien unter `/etc/pve` sofort auf allen Nodes sichtbar

Intervalle

Kurzes Intervall (1 Min.)

kleines RPO
mehr Last/Bandbreite

Langes Intervall (z. B. 1 h)

größeres RPO
weniger Last/Bandbreite

Faustregel: nicht „so kurz wie möglich“, sondern „so kurz wie nötig“ – das gewünschte RPO bestimmt das Intervall, nicht die technische Untergrenze.

- Minimal ein Minuten-Intervall möglich, Zeitplan über systemd-Calendar-Syntax (z. B. `*/*` für alle 5 Minuten, stündlich oder täglich)
- Kürzeres Intervall bedeutet kleineres RPO – ein 1-Minuten-Takt begrenzt den maximalen Datenverlust im Ernstfall auf rund eine Minute
- Aber: mehr Last – jeder Lauf legt Snapshots an und beansprucht kurzzeitig CPU und I/O auf Quelle wie Ziel; bei VMs mit hoher Schreibrate summiert sich das spürbar
- Für ruhige VMs (wenig Schreiblast, z. B. interne Dokuserver) reichen oft 15- oder 30-Minuten-Intervalle, für kritische Systeme wie Datenbank oder ERP eher 1–5 Minuten

RPO-Berechnung

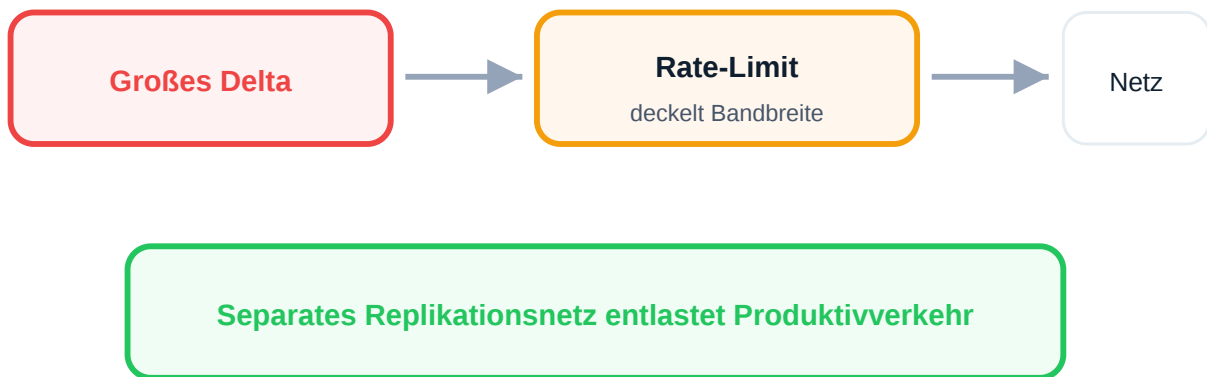
$$\text{Delta} \div \text{Netzdurchsatz} = \text{Übertragungsdauer}$$

Übertragungsdauer muss kleiner sein als das Intervall

Max. Datenverlust $\approx$ Zeit seit letzter Replikation

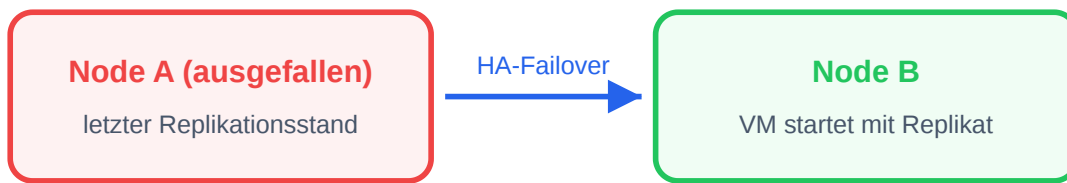
- Maximaler Datenverlust $\approx$ Zeit seit der letzten erfolgreichen Replikation
- $\text{Delta} \div \text{Netzdurchsatz} = \text{Übertragungsdauer}$
- Die Übertragungsdauer muss ins gewählte Intervall passen, sonst läuft die Replikation der VM hinterher und die Läufe stapeln sich
- Beispiel: 5 GB Delta über ein 1-Gbit-Netz (effektiv ca. 100 MB/s) ergeben rund 50 Sekunden Übertragungsdauer – passt bequem in ein 1-Minuten-Intervall. Bei 10 GB Delta wären es schon gut 100 Sekunden – das Intervall würde nicht mehr eingehalten
- Bleibt die Übertragung regelmäßig zurück: Intervall ehrlich verlängern oder die Netzanbindung beschleunigen (z. B. von 1 GbE auf 10 GbE)

Bandbreite & Last



- Ein großes Delta braucht ein entsprechend schnelles Netz
- Rate-Limit (`--rate` bei `pvesr`) deckelt die Replikations-Bandbreite in MB/s und schützt das Produktivnetz vor Sättigung während eines großen Laufs
- Ein separates Replikationsnetz (eigene NIC/VLAN, ausschließlich für Storage- und Replikationsverkehr) ist bei größeren Deltas oder vielen replizierten VMs sinnvoll, damit ein einzelner großer Lauf nicht den VM-Netzverkehr ausbremst
- Zum Einordnen: 1 GbE liefert real ca. 100–110 MB/s, 10 GbE entsprechend rund 1–1,1 GB/s – bei mehreren TB replizierten VMs macht das den Unterschied zwischen Minuten und Sekunden pro Lauf
- Für kleine Setups mit wenigen, kleinen VMs reicht das normale Cluster-Netz meist aus – ein eigenes Replikationsnetz lohnt sich erst ab spürbarer Zusatzlast

Zusammenspiel mit HA

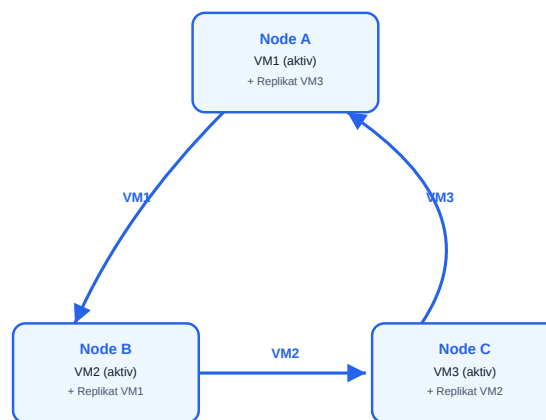


Günstige Alternative zu Ceph – kleines, kalkulierbares RPO

Für den typischen KMU-Fall (2 Nodes, überschaubares Budget) ist lokales ZFS mit Replikation oft die pragmatischste Kombination aus Ausfallsicherheit und Aufwand – Ceph liefert Datenverlust nahe null, aber zu einem deutlich höheren Infrastruktur- und Betriebsaufwand.

- Replikation macht lokalen Storage überhaupt erst HA-fähig – ohne sie hätte ein Ersatz-Node nach einem Ausfall schlicht keine Daten
- Failover nutzt den zuletzt replizierten Stand – RPO und Replikations-Intervall hängen deshalb direkt zusammen (1-Minuten-Intervall = maximal 1 Minute Datenverlust im Failover-Fall)
- Günstige Alternative zu Ceph – kleines, kalkulierbares RPO bereits auf 2 Nodes plus QDevice (siehe Modul 8), statt mindestens 3 Nodes und eigenem Storage-Netz für Ceph
- Nach einem Failover ist der ausgefallene Node im Rückstand – kommt er zurück, sollte der Replikationsstand kontrolliert und die Richtung ggf. neu aufgesetzt werden, bevor die VM dorthin zurückwandert

ZFS-Replikation im 3-Node-Cluster



Replikationsziele bewusst über die Nodes verteilt –
nicht alle VMs auf denselben Partner-Node

Entscheidungshilfe für den 3-Node-Cluster: Wer ein höheres, aber kalkulierbares RPO im Minutenbereich akzeptiert und Aufwand sparen will, plant die Replikationsziele gezielt und verteilt über ZFS-Replikation. Wer ein RPO nahe null und automatische Verteilung über den ganzen Cluster braucht, investiert den höheren Aufwand in Ceph.

ZFS-Replikation

RPO im Minutenbereich, geringerer Aufwand bei Aufbau und Betrieb, gezielte 1:1-Verteilung planbar

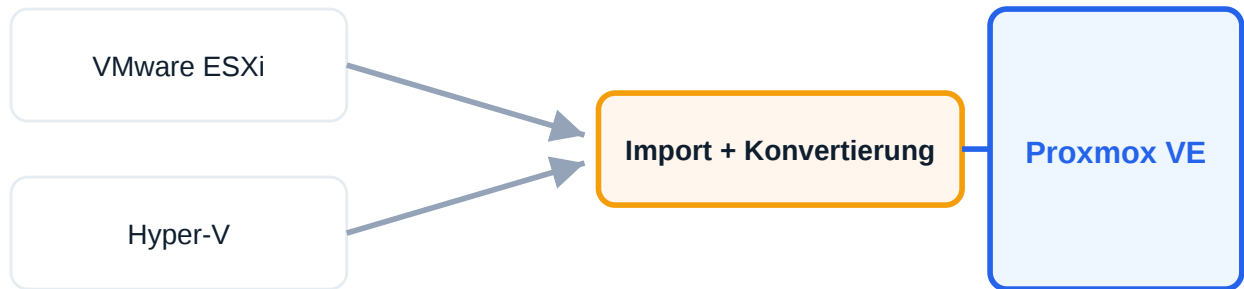
Ceph

RPO praktisch 0, mehr Aufwand bei Aufbau und Betrieb, automatische Verteilung über den ganzen Cluster

- Replikation ist 1:1 gerichtet, nicht verteilt wie Ceph – es gibt kein automatisches Verteilen über den ganzen Cluster
- Pro VM wird das bzw. werden die Ziele explizit festgelegt; Mehrfachziel ist möglich, z. B. repliziert eine VM auf Node A sowohl zu Node B als auch zu Node C
- Best Practice: jede VM auf mindestens einen definierten Partner-Node replizieren – die Ziele dabei bewusst über die Nodes verteilen, nicht alle VMs auf denselben Partner-Node
- Grenze: Ein Failover ist nur auf Nodes möglich, die tatsächlich eine aktuelle Replik der betroffenen VM besitzen – ein Node ohne Replik kann die VM im Ausfallfall nicht übernehmen

M10 · Migration & vDisk-Konvertierung

Von VMware und Hyper-V zu Proxmox – der Hauptgrund für den Umstieg



Vorbereitung

- ✓ Guest-Tools der Altplattform entfernen
- ✓ VirtIO-Treiber bereitlegen
- ✓ Netz, MAC, Boot-Reihenfolge dokumentieren

Migration ist zu 90 % Vorbereitung und nur zu 10 % Knopfdruck. Wer die Reihenfolge umdreht, verbringt die Nacht mit Fehlersuche statt mit einem sauberen Umzug.

Software

Guest-Tools der Altplattform entfernen, VirtIO-ISO bereitlegen

Netzwerk

MAC-Adresse, VLAN, IP und DNS vorher dokumentieren

Lizenzen

Produktschlüssel und Aktivierungsstatus vor dem Umzug sichern

- Guest-Tools der Altplattform entfernen (VMware Tools, Hyper-V-Integrationsdienste) – alte Tools führen nach dem Umzug sonst leicht zu Bluescreens oder falsch erkannter virtueller Hardware
- Bei Windows zusätzlich vorab den Product Key sichern, z. B. mit `wmic path softwarelicensingservice get OA3xOriginalProductKey` für einen im BIOS/UEFI eingebetteten OEM-Key – nach der Migration ist dieser Key oft nicht mehr auslesbar
- VirtIO-Treiber bereitlegen und, wo die Altplattform es zulässt, schon vor der Migration im Gastsystem einspielen
- Netz, MAC-Adressen, VLAN, IP-Konfiguration, DNS-Einträge und Boot-Reihenfolge vorher dokumentieren – wichtig, weil DHCP-Reservierungen und Firewall-Regeln oft an die MAC-Adresse gebunden sind

VMware → Proxmox

Import-Assistent: ESXi-Host hinzufügen

Host-Adresse

esxi01.kunde.local

Zugangsdaten

root ·

Gefundene VMs

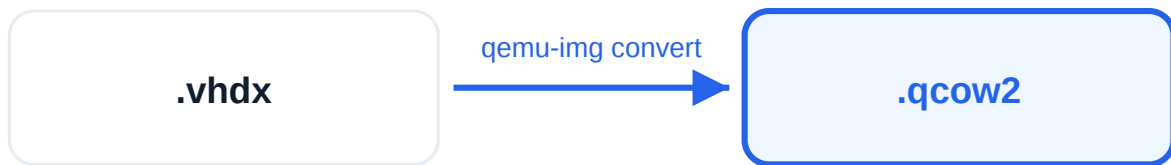
☒ FIBU-Server (120 GB)

☐ Terminalserver (250 GB)

Der Import ist der Umzugswagen, nicht das fertig eingerichtete neue Zuhause – Boot-Modus, VirtIO-Treiber und Guest Agent sind eigene Arbeitsschritte danach, dazu mehr in den folgenden Blöcken.

- Eingebauter ESXi-Import-Assistent (ab Proxmox VE 8.2): Zugangsdaten des ESXi-Hosts (oder vCenter) eingeben, Proxmox listet die VMs direkt auf und liest Festplatten sowie Konfiguration aus
- Direkter Import vom ESXi-Host übers Netzwerk, ohne Umweg – deutlich weniger Handarbeit als in älteren Proxmox-Versionen, die zwingend einen OVF/OVA-Export voraussetzten
- Alternativ: Import über OVF/OVA-Paket – sinnvoll bei sehr alten ESXi-Versionen ohne API-Zugriff oder wenn getrennt vom Quellsystem gearbeitet werden soll
- Für Skript- oder Batch-Migrationen alternativ `virt-v2v` (aus libguestfs): konvertiert VMware- und Hyper-V-VMs inklusive Treiber-Anpassung im Gastsystem in einem Rutsch

Hyper-V → Proxmox



Generation-2-VM (UEFI) beim Boot-Modus beachten

Bei aktivem BitLocker auf der Quell-Disk vorher entschlüsseln oder den Wiederherstellungsschlüssel sichern – kopieren lässt sich die Disk zwar, ohne den ursprünglichen TPM-Schlüssel bleibt sie nach der Migration aber verschlossen.

- Kein so direkter Import-Assistent wie bei ESXi – der Kern der Arbeit ist die Konvertierung: Hyper-V speichert Disks als VHD/VHDX, Proxmox bevorzugt qcow2 oder raw
- Werkzeug: `qemu-img` – derselbe Befehl wandelt auch das VMware-Format vmdk um, ein Universalübersetzer für praktisch alle gängigen vDisk-Formate
- Guest-Betriebssystem so gut wie möglich vorab auf VirtIO vorbereiten, etwa durch frühzeitiges Einspielen der Treiber im Altsystem
- Generation-2-Hyper-V-VMs booten per UEFI, Generation-1-VMs klassisch per BIOS – der Boot-Modus auf Proxmox muss zur Quelle passen, sonst startet die VM nicht
- Dynamische VHDX-Disks (die mitwachsen) lassen sich genauso konvertieren wie fixe – `qemu-img` erkennt das Format automatisch, eine manuelle Umwandlung in Hyper-V vorher ist nicht nötig

vDisk-Konvertierung mit qemu-img

```
$ qemu-img convert -O raw quelle.vmdk ziel.raw
```

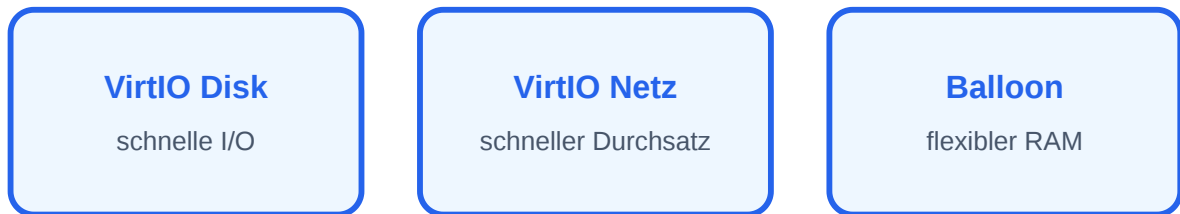
```
(100.00/100%)
```

fertig – ziel.raw bereit für Proxmox-Storage

Ein Befehl für vmdk, vhdx und viele weitere Formate

- Grundbefehl: `qemu-img convert -p -O qcow2 quelle.vmdk ziel.qcow2` – `-p` zeigt den Fortschritt, `-O` legt das Zielformat fest
- Unterstützte Quellformate u. a. vmdk (VMware), vhd/vhdx (Hyper-V) und vdi (VirtualBox) – alle laufen über denselben Befehl auf qcow2 oder raw
- Zielformat nach Storage wählen: raw für ZFS/LVM-thin (der Storage übernimmt selbst Thin-Provisioning und Snapshots), qcow2 für Verzeichnis-Storage (eigene Snapshot-Fähigkeit, etwas mehr Overhead als raw)
- Nach der Konvertierung wird die Disk mit `qm importdisk <vmid> ziel.qcow2 <storage>` einer bereits angelegten, leeren VM-Hülle zugeordnet und im Hardware-Dialog als SCSI/VirtIO-Disk aktiviert
- qcow2 bleibt sparse (nur genutzte Blöcke belegen Platz); raw ist das auf ZFS-Zvols durch deren Thin-Provisioning ebenfalls, auf klassischen Dateisystemen dagegen nicht automatisch

VirtIO-Treiber (Windows)



Ohne VirtIO: langsam oder gar kein Boot

Henne-Ei-Problem: Windows kann nicht von einer VirtIO-SCSI-Disk booten, solange der Treiber dafür fehlt. Deshalb beim ersten Boot zunächst als emulierte SATA/IDE-Disk starten, den VirtIO-Treiber von der ISO installieren und erst danach in der VM-Hardware auf VirtIO SCSI umstellen und neu starten.

- Ohne VirtIO: spürbar langsamerer Betrieb durch IDE/SATA-Emulation oder im ungünstigsten Fall gar kein Boot, wenn die Disk direkt als VirtIO-SCSI angesprochen wird
- Treiber-ISO (offizielles VirtIO-Windows-Repository) als zweites CD-Laufwerk einbinden und die Treiber über den Geräte-Manager laden
- Disk- (`viostor` / `vioscsi`), Netz- (`NetKVM`) und Balloon-Treiber (`Balloon`, für flexiblen Arbeitsspeicher) gehören zum Kern-Set, dazu `qx1` für Grafik und `vioserial` für die Guest-Agent-Kommunikation
- Vor dem Umstieg auf VirtIO SCSI (oder anderen riskanten Migrationsschritten) einen **Snapshot** der VM anlegen – schnelle Absicherung, um bei Problemen sofort zurückzuspringen; kein Ersatz für ein Backup, da der Snapshot auf demselben Storage liegt wie die VM selbst

QEMU Guest Agent

**Sauberes
Herunterfahren**

**IP-Anzeige
in der GUI**

Backup-Freeze
konsistente Backups

- Sauberes Herunterfahren direkt aus der Proxmox-Oberfläche, statt eines harten Ausschaltens, das bei laufenden Schreibvorgängen Datenverlust riskiert
- Zeigt die IP-Adresse der VM in der GUI an – ohne Agent bleibt nur Raten oder ein Blick ins DHCP-Log
- Sorgt für konsistente Backups durch kurzes Einfrieren des Dateisystems (Freeze) während des Snapshots – besonders wichtig bei Datenbank-VMs
- Installation unter Windows über das VirtIO-Treiber-ISO (enthält den Installer `qemu-ga-x86_64.msi`), unter Linux meist als Paket `qemu-guest-agent` aus den Standard-Repositories
- In der VM-Hardware muss die Option „QEMU Guest Agent“ zusätzlich aktiviert werden – sonst wartet Proxmox beim Booten unnötig auf eine Antwort, die nie kommt, was den Start spürbar verzögert

UEFI/BIOS & TPM

BIOS (SeaBIOS)

klassische, ältere VMs

UEFI (OVMF)

Hyper-V Gen. 2, moderne VMs

+ TPM 2.0 für Windows 11

BitLocker-verschlüsselte Windows-Disks sind an das ursprüngliche TPM gebunden. Das neue virtuelle TPM in Proxmox ist ein anderes Gerät – vorher entschlüsseln oder den Wiederherstellungsschlüssel griffbereit halten, sonst hängt die VM beim Boot in der BitLocker-Wiederherstellung fest.

- Boot-Modus auf Proxmox muss zur Quelle passen: SeaBIOS für klassische BIOS-VMs, OVMF für UEFI – Hyper-V-Generation-2 und die meisten modernen VMware-VMs sind UEFI
- Windows 11 benötigt zusätzlich ein virtuelles TPM 2.0, das Proxmox als eigenes Gerät (`tpmstate0`) bereitstellt – ohne TPM verweigert Windows 11 teils Updates oder startet gar nicht erst
- Falscher Boot-Modus zeigt meist nur einen schwarzen Bildschirm ohne Fehlermeldung – nicht raten, sondern vorher prüfen, wie die Quell-VM tatsächlich gebootet hat
- Bei OVMF zusätzlich eine kleine EFI-Disk (`efidisk0`) für die VM anlegen – dort liegen die UEFI-Variablen wie Boot-Reihenfolge; ohne diese Disk vergisst die VM ihre Boot-Konfiguration bei jedem Neustart

Windows-Reaktivierung

⚠ Windows-Aktivierung

Hardware-ID ändert sich → Neuaktivierung möglich

Windows- und App-Lizenzen vorher prüfen

Vorher: Lizenzstatus dokumentieren

Nichts ist unangenehmer, als beim Kunden vor Ort nach einem verlorenen Produktschlüssel zu suchen, während die Fachsoftware auf die Freischaltung wartet – das Lizenz-Inventar gehört fest in die Vorbereitungscheckliste.

- Geänderte virtuelle Hardware (CPU-ID, Mainboard-Kennung, Netzwerkkarte) lässt Windows glauben, es liefe auf einem neuen Rechner – eine Neuaktivierung wird dann fast immer verlangt
- Digital-Entitlement-Aktivierungen (an das Microsoft-Konto gebunden) lassen sich meist per Online-Assistent oder erneutes Anmelden mit dem Microsoft-Konto reaktivieren; im BIOS der alten Hardware eingebrannte OEM-Keys gehen dagegen mit der virtuellen Hardware verloren
- Windows- und Fachanwendungs-Lizenzen vorab prüfen – auch Branchensoftware bindet sich teils an Hardware-Merkmale wie Datenträger-Seriennummer, MAC-Adresse oder CPU-ID
- Lizenzstatus vor der Migration dokumentieren (Produktschlüssel, Aktivierungsart, ggf. Kontobindung) – siehe Checkliste im Vorbereitungs-Block

USB-/Lizenz-Dongle



Alternativ: USB-über-Netzwerk-Adapter

- Dongle per USB-Passthrough direkt in die VM durchreichen – Proxmox kann USB-Geräte nach Port oder nach fester Vendor-/Produkt-ID durchreichen; die ID-Variante übersteht auch einen anderen USB-Port oder einen Hostwechsel
- Oft entscheidend für KMU-Fachsoftware (Buchhaltung, Branchenlösungen, CAD) – ohne funktionierenden Dongle-Durchgriff ist eine Migration für diese Anwendungen schlicht nicht möglich
- Alternativ: USB-über-Netzwerk-Lösungen mit kleinen Hardware-Adaptern, sinnvoll wenn der Dongle nicht direkt am Proxmox-Host, sondern z. B. an einem Arbeitsplatz steckt
- Passthrough nach Vendor-/Produkt-ID in der VM-Hardware eintragen (z. B. `usb0: host=1234:5678`) statt nach Bus-/Port-Nummer – Port-Zuordnungen ändern sich gerne beim nächsten Hardware-Tausch

Konsole als Rettungsanker

noVNC – Konsole: FIBU-Server

```
Proxmox VE – SeaBIOS (version rel-1.16.0)
```

```
Booting from Hard Disk...
```

```
sichtbar auch ohne Netzwerk/Grafiktreiber in der VM
```

- noVNC (Standard, direkt im Browser) oder SPICE (mehr Komfort: Zwischenablage, dynamische Auflösung, benötigt einen kleinen Client) sind echte Hypervisor-Konsolen, keine Fernwartung innerhalb der VM
- Zeigen BIOS/UEFI-Bildschirm, Bootvorgang und sogar den abgesicherten Modus, unabhängig davon, ob in der VM Netzwerk oder ein funktionierender Grafiktreiber vorhanden ist
- Nach einem Treiberfehler ist man dadurch nie ausgesperrt, weil die Konsole auf Hypervisor-Ebene arbeitet und nicht von einem Dienst innerhalb der VM abhängt
- Genau deshalb ist die Konsole der Rettungsanker beim VirtIO-Umstieg: Klappt der Wechsel auf VirtIO-SCSI nicht, zeigt die Konsole den Bluescreen oder die Fehlermeldung, während eine RDP- oder SSH-Verbindung in dem Moment schon tot wäre

Nachkontrolle



Netz: MAC, VLAN, IP prüfen



Boot-Reihenfolge, alle Dienste gestartet



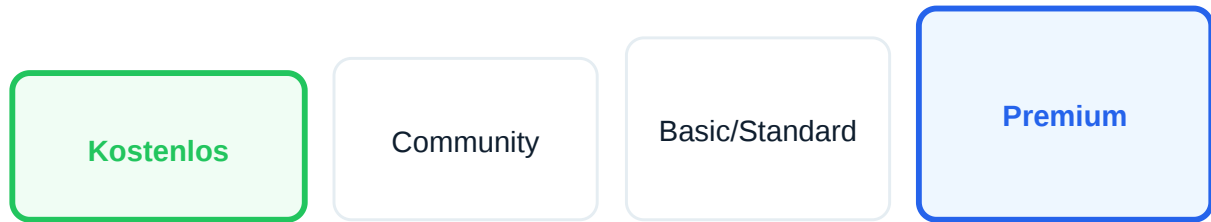
Performance & Treiber final checken

Migration endet nicht beim ersten erfolgreichen Login, sondern erst, wenn die komplette Checkliste grün ist – „bootet, passt schon“ ist keine Endabnahme.

- Netz prüfen: MAC-Adresse mit der Dokumentation aus der Vorbereitung abgleichen, VLAN-Zuordnung, IP-Adresse und DNS-Auflösung
- Boot-Reihenfolge und alle Dienste kontrollieren – insbesondere automatisch startende Anwendungen, die nach dem Umzug tatsächlich wieder hochkommen müssen
- Performance und Treiber final checken (VirtIO aktiv? Guest Agent meldet sich? Balloon-Treiber aktiv?), bevor „fertig“ gemeldet wird
- Backup-Job für die migrierte VM einrichten oder anpassen – eine frisch migrierte VM ohne eigenes Backup ist ein Restrisiko, das schnell vergessen wird

M11 · Lizenzierung

Kostenlos vs. Subscription – das Support-Modell realistisch einordnen



Steigender Support-Umfang, alle Stufen technisch gleich funktionsfähig

Kostenlos = voll funktionsfähig



Cluster

Keine künstliche Feature-Sperre



Hochverfügbarkeit (HA)



Backup



Migration

Der Unterschied zwischen kostenloser Nutzung und Subscription liegt ausschließlich beim Update-Kanal und beim Support-Anspruch – nicht bei der Funktionalität.

- Alle Features auch ohne Lizenz nutzbar – Cluster, Hochverfügbarkeit (HA), Live-Migration, Backup inklusive Proxmox Backup Server, Replikation sowie Software-Defined Storage (ZFS, Ceph) und Software-Defined Networking
- Keine künstliche Feature-Sperre zwischen einer abgespeckten „Community“- und einer vollwertigen „Enterprise“-Variante, wie es bei anderen Virtualisierungs-Herstellern üblich ist
- Ein produktiver Mehr-Node-Cluster mit HA lässt sich technisch vollständig ohne einen Cent Lizenzgebühr aufbauen und dauerhaft betreiben
- Praktische Konsequenz: Evaluierung, Testlabor und auch der produktive Kleinbetrieb sind ohne Kaufentscheidung möglich – die Subscription kommt erst ins Spiel, wenn Update-Stabilität und Herstellersupport wichtiger werden als die gesparte Lizenzgebühr

Was die Subscription bringt



- Zugang zum Enterprise-Repository (`pve-enterprise`) mit Paketen, die vor der Freigabe zusätzlich getestet und über einen längeren Zeitraum validiert wurden
- Support-Tickets im Kundenportal, gestaffelt nach gebuchtem Tier – mehr Tickets und kürzere Reaktionszeiten in den höheren Stufen
- Mehr Stabilität für den Produktivbetrieb, besonders relevant bei Servern mit Kernsystemen wie Datenbanken oder ERP, wo ein fehlerhaftes Update im schlimmsten Fall den Geschäftsbetrieb unterbricht
- Wichtig für die Erwartungshaltung: bezahlt wird nicht für Funktionen, sondern für Absicherung – als Faustregel lohnt sich mindestens die Basic-Stufe für jeden unternehmenskritischen Produktivserver, während Testlabore meist ohne Subscription auskommen

Subscription-Tiers

STUFE	SCHWERPUNKT
Community	Enterprise-Repo-Zugang
Basic	Ticket-Support, moderate Reaktionszeit
Standard	mehr Tickets, schnellere Reaktion
Premium	meiste Tickets, schnellste Reaktion
Abrechnung: pro CPU-Sockel pro Jahr	

Community

Einstiegsstufe – zentraler Nutzen ist der Zugang zum Enterprise-Repository, kein garantierter Ticket-Support

Basic

Ticket-Support mit moderater Reaktionszeit, sinnvoller Einstieg für produktive KMU-Server

Standard

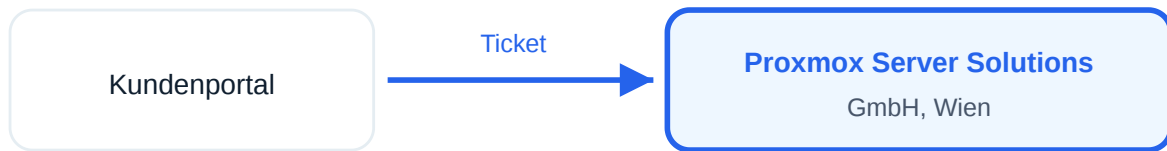
Mehr Tickets pro Jahr und schnellere Reaktionszeit, für Umgebungen mit mehreren produktiven Nodes

Premium

Höchste Ticket-Anzahl und schnellste Reaktionszeit, für unternehmenskritische Cluster mit wenig Toleranz für Ausfallzeiten

- Abrechnung pro CPU-Sockel pro Jahr – ein Server mit zwei CPUs zahlt für zwei Sockel-Lizenzen, unabhängig von der Kernzahl je Sockel
- Alle vier Stufen schalten dasselbe Enterprise-Repository frei – der Unterschied liegt ausschließlich im Umfang des Ticket-Supports

Hersteller & Support-Kanal



Ticket-basiert, kein 24/7-Telefon-Support

Bewusst anderes Modell als bei den großen Virtualisierungs-Konzernen: Proxmox konzentriert sich auf Software und getestete Updates, die eigentliche Rund-um-die-Uhr-Betreuung liegt beim IT-Dienstleister vor Ort.

- Hersteller: Proxmox Server Solutions GmbH mit Sitz in Wien, Österreich – ein EU-Unternehmen, kein Konzern außerhalb der EU
- Für deutschsprachige Kunden praktisch: deutschsprachiger Kontakt und EU-Datenschutzrecht statt eines Software-Konzerns in Übersee
- Support läuft ticket-basiert über das Kundenportal – Reaktionszeit und Ticket-Kontingent richten sich nach dem gebuchten Tier
- Kein 24/7-Telefon-Support direkt vom Hersteller – wer nachts um drei anruft, erreicht bei Proxmox selbst niemanden

Repos richtig setzen

Repositories

✗ **pve-enterprise**

deaktiviert (keine Subscription)

✓ **pve-no-subscription**

aktiviert

- Enterprise-Repo (`/etc/apt/sources.list.d/pve-enterprise.list`) nur mit gültiger Subscription aktiv lassen
- Ohne Subscription: No-Subscription-Repo (`pve-no-subscription`) aktivieren – enthält dieselben Pakete, nur ohne die zusätzliche Enterprise-Testphase
- Falsche Repo-Wahl führt zu Update-Fehlern: `apt update` meldet bei aktivem Enterprise-Repo ohne gültige Subscription typischerweise einen `401`-Fehler beim Abruf der Paketlisten
- Dieser Schritt gehört zu den ersten Handgriffen direkt nach einer Neuinstallation, noch vor dem ersten Update-Lauf

Erwartung gegenüber VMware

VMware-Gewohnheit

Rundum-24/7 vom Hersteller

hohe Lizenzkosten

Proxmox + Dienstleister

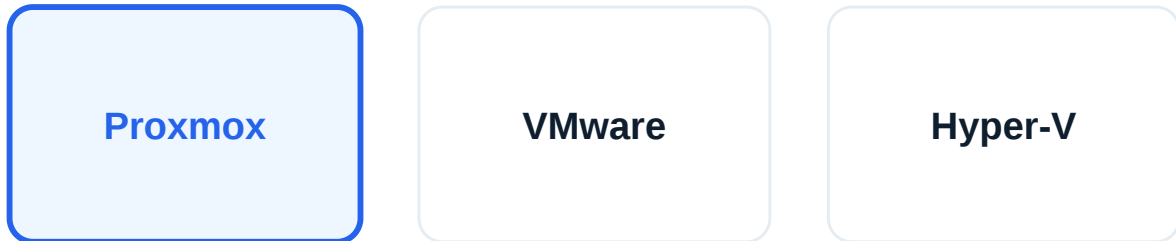
Dienstleister als Support-Ebene

deutlich niedrigere Kosten

- Kein Rundum-24/7-Support direkt vom Hersteller, wie es viele Kunden aus der VMware-Welt gewohnt sind
- Der Dienstleister übernimmt die eigentliche Support-Ebene – ein Ansprechpartner, der die konkrete Umgebung kennt, statt eines anonymen Ticket-Systems
- Dafür deutlich niedrigere Lizenzkosten – das eingesparte Budget lässt sich gezielt in guten lokalen Support investieren, statt es an den Hersteller abzuführen
- Für viele KMU geht diese Rechnung auf: weniger Lizenzkosten, dafür ein Mensch, der reagiert, sobald es wirklich brennt

M12 · Vergleich & Kosten

Stärken, Schwächen, Kosten – ehrliche Einordnung für die Entscheidung



Stärken, Schwächen, Kosten – ehrliche Einordnung

Stärken von Proxmox

- ✓ Kostenlos voll funktionsfähig, kein Lock-in
- ✓ Offen (Debian/KVM), volle Kontrolle
- ✓ Cluster, HA, Backup integriert

- Kostenlos voll funktionsfähig, kein Hersteller-Lock-in – Wechsel des Dienstleisters oder Eigenbetrieb ist jederzeit möglich, ohne an eine Lizenz gebunden zu sein
- Offen (Debian/KVM) – volle Kontrolle über das System, Konfiguration liegt als lesbare Textdateien vor statt in einer Blackbox
- Cluster, HA und Backup bereits integriert, nicht als teures Zusatzprodukt dazugekauft wie bei manchen Wettbewerbern
- Aktive Weiterentwicklung mit kurzen Release-Zyklen und einer Community, die seit dem Broadcom-Umbau bei VMware spürbar gewachsen ist

Schwächen von Proxmox



Kein 24/7-Enterprise-Support ab Werk



Kleineres Ökosystem/Marktplatz



Mehr Linux-Eigenverantwortung

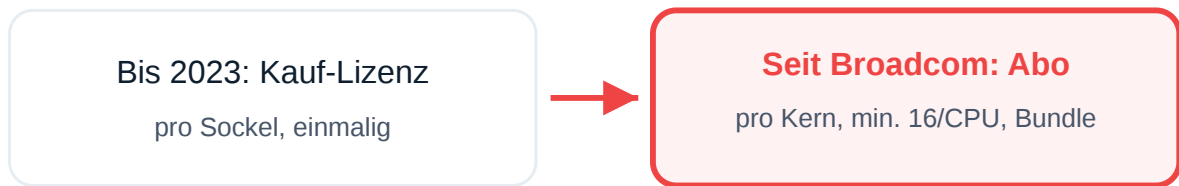
- Kein 24/7-Enterprise-Support ab Werk – diese Rolle übernimmt, wie in Modul 11 beschrieben, der Dienstleister
- Kleineres Ökosystem und Marktplatz als bei VMware – weniger fertige Drittanbieter-Integrationen, Backup-Tools und Monitoring-Plugins von der Stange
- Mehr Linux-Eigenverantwortung im eigenen Haus nötig – Terminal- und Debian-Kenntnisse helfen deutlich, eine reine Klick-Oberfläche wie bei einer geschlossenen Appliance ersetzt das nicht
- Für Großkonzerne mit strengen Enterprise-Supportverträgen und Spezial-Ökosystem-Anforderungen bleibt VMware oft gesetzt – im klassischen KMU-Umfeld sind diese Einschränkungen meist zweitrangig

Feature-Vergleich

KRITERIUM	PROXMOX	VMWARE	HYPER-V
HA / Live-Migration	ja, Standard	ja, Standard	ja, Standard
Backup / Snapshots	PBS integriert	Snapshot; Backup extra	Windows-Bordmittel
Ökosystem	klein, wachsend	sehr groß	groß (Microsoft)
Preismodell (Basis)	kostenlos	Abo/Kern, Pflicht	in Windows Server
Support vom Hersteller	Ticket, kein 24/7	24/7 möglich (teuer)	MS-Support
Lock-in	gering (offen)	hoch (Bundle)	mittel
Windows-Integration	gut (VirtIO)	gut	am besten
Blau hervorgehoben: jeweils führende Plattform beim Kriterium			

- HA und Live-Migration: alle drei Plattformen beherrschen es im Kern gleich gut, der Admin-Alltag fühlt sich ähnlich an
- Backup: Proxmox bringt mit dem Proxmox Backup Server eine integrierte Lösung mit, bei VMware und Hyper-V braucht es meist ein zusätzliches Drittprodukt (z. B. Veeam)
- Ökosystem und Drittanbieter-Integrationen: VMware führt, jahrzehntelanger Marktführer-Vorsprung bei Zertifizierungen und Partnerprodukten
- Offenheit und Preis: Proxmox führt – kein Lock-in, kein Abozwang, volle Kontrolle über das System
- Windows-Integration: Hyper-V führt in einer reinen Microsoft-Umgebung, nahtlos ins Active Directory eingebunden

VMware unter Broadcom



Marktbeobachter berichten teils ein Mehrfaches der früheren Kosten

(Einzelangaben je nach Anbieter/Deal-Größe sehr unterschiedlich)

- Nur noch Abo, keine Kauf-Lizenzen mehr – Abrechnung pro Kern, mit einem Minimum von 16 Kernen pro CPU-Sockel, selbst wenn die CPU tatsächlich weniger Kerne hat
- Bundle-Zwang (VCF/VVF) statt Einzelprodukten – wer bisher nur vSphere brauchte, kauft jetzt ein großes Paket mit Funktionen, die im KMU oft ungenutzt bleiben
- Dazu Lizenz-Audits bei Altkunden, die von den früheren Perpetual-Lizenzen auf das neue Abomodell umsteigen müssen oder sollen
- Laut Marktbeobachtern reduziert die Mehrheit der Firmen ihren VMware-Anteil aktiv (Stand 2026) – ein Vertrauensverlust, der viele der in Modul 14 beschriebenen Migrationsprojekte erst ausgelöst hat

Hyper-V-Kosten

Windows Server Standard

2 VMs pro Lizenz
mehr VMs = mehrfach lizenzieren

Windows Server Datacenter

unbegrenzt viele VMs
ca. 423 \$/Kern (Listenpreis 2026)

- Hyper-V selbst ist im Windows Server enthalten – kein separates Hypervisor-Produkt und keine eigene Hypervisor-Lizenzgebühr
- Windows Server **Standard** erlaubt nur 2 VMs pro Lizenz – für mehr VMs muss dieselbe Lizenz mehrfach erworben werden (jeweils +2 VMs)
- Für viele VMs braucht es die **Datacenter**-Lizenz mit unbegrenzt vielen VMs (ca. 423 \$/Kern Listenpreis, Stand 2026)
- Die Windows-Lizenzkosten für Host und Gäste bleiben in jedem Fall bestehen – wer überwiegend Linux-Gäste betreibt, zahlt für eine Windows-Lizenz, die er eigentlich nur wegen des Hypervisors braucht

Proxmox vs. Hyper-V im direkten Vergleich

KRITERIUM	PROXMOX VE	HYPER-V
Linux / Container	LXC nativ + KVM	nur volle VM
Windows / AD-Integration	gut (VirtIO)	nahtlos, System Center
Gast-Lizenzkosten	separate OS-Lizenz	oft in Datacenter dabei
Hypervisor-Lizenz	kostenlos	in Windows Server
Backup	PBS integriert	Bordmittel / Azure Backup
Cluster & HA	ja, Standard	ja, Failover-Cluster-Manager
Verwaltung	Web-GUI / CLI / API	Manager / PowerShell / SC
Lernkurve (Windows-Admin)	Linux-Grundlagen nötig	vertraut, keine Umstellung

Blau hervorgehoben: jeweiliger Vorteil bei diesem Kriterium
Beide beherrschen Cluster/HA im Kern gleich gut – der Unterschied liegt im Ökosystem und in der bevorzugten Gast-Welt

Entscheidungshilfe: Überwiegend Linux-Workloads, Wert auf Kostenkontrolle und keine Lust auf Windows-Lizenzpflicht pro Host → Proxmox VE. Reine oder überwiegende Windows-Server-Landschaft mit vorhandenen Datacenter-Lizenzen, Windows-Admin-Team ohne Linux-Erfahrung und Wunsch nach durchgängigem Microsoft-Support → Hyper-V bleibt die pragmatischere Wahl. Gemischte Umgebungen lohnt es, anhand der TCO-Rechnung im nächsten Block konkret durchzurechnen statt nach Bauchgefühl zu entscheiden.

Für Proxmox spricht

LXC-Container nativ, kostenloser Hypervisor, integriertes Backup (PBS), kein Windows-Lizenzzwang für Linux-Gäste, offene Architektur ohne Lock-in

Für Hyper-V spricht

nahtlose Active-Directory-Integration, vertraute Verwaltung für Windows-Admins (PowerShell, System Center), oft schon mitlizenziert bei vorhandener Windows-Server-Datacenter-Lizenz, durchgängiger Microsoft-Support

Kostenvergleich

Proxmox		optional, ab 120 €/Sockel/Jahr
VMware (VVF/VCF)		ca. 150–350 €/Kern/Jahr (Marktangaben, kein Listenpreis öffentlich)
Hyper-V (Datacenter)		ca. 423 \$/Kern (Windows-Server-Lizenz, Listenpreis 2026)

Proxmox-Zahl: offizielle Preisliste. VMware/Hyper-V: Markt- bzw. Listenpreise, Stand 2026

- Proxmox: optional, offizielle Preisliste ab 120 €/Sockel/Jahr (Community-Tier) – ein Server mit einem Sockel kostet im Jahr damit etwa so viel wie eine einzelne halbtägige Dienstleister-Rechnung
- VMware: Abo pro Kern, laut Marktangaben ca. 150–350 €/Kern/Jahr, dazu das 16-Kern-Minimum pro Sockel – ein Zwei-Sockel-Server mit modernen CPUs kommt so schnell auf einen fünfstelligen Jahresbetrag
- Hyper-V: an die Windows-Server-Lizenz gekoppelt, kein separates Hypervisor-Abo, dafür Datacenter-Lizenzkosten pro Kern plus laufende Windows-Gastlizenzen
- Wichtig beim Lesen der Zahlen: die Proxmox-Angabe ist der offizielle Listenpreis, die VMware- und Hyper-V-Werte sind Markt- bzw. Listenpreise (Stand 2026) – reale Angebote können je nach Deal-Größe und Vertragslaufzeit abweichen

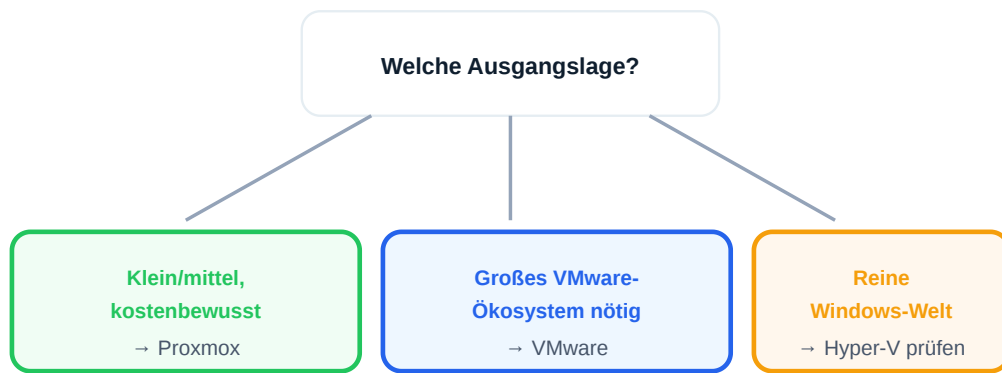
TCO für KMU



Proxmox im KMU meist niedrigste TCO

- Lizenz, Support und laufenden Betrieb zusammenrechnen, nicht nur den Listenpreis – eine günstige Lizenz mit hohem Betriebsaufwand kann in Summe teurer sein als eine teurere Lösung mit wenig Pflegeaufwand
- Migrationsaufwand als einmalige Position einplanen: Planung, Testphase, tatsächliche Umzugszeit und ein Sicherheitspolster für Nacharbeiten
- Beispielrechnung: bei den zuvor genannten VMware-Preisen amortisiert sich ein einmaliger Migrationsaufwand oft schon innerhalb von ein bis zwei Jahren gegenüber den weiterlaufenden VMware-Abokosten
- Proxmox liegt im KMU-Umfeld meist bei der niedrigsten Gesamt-TCO, selbst wenn man realistisch etwas mehr Eigenaufwand für den laufenden Betrieb einplant
- Konkret für das KMU-Referenz-Szenario dieses Lernbereichs (1–2 Nodes, 4–6 VMs, siehe Modul 0): selbst mit Proxmox-Subscription für beide Nodes (Community-Tier, ca. 120 €/Sockel/Jahr) bleibt die jährliche Lizenzsumme im niedrigen dreistelligen Bereich – ein Bruchteil dessen, was allein das 16-Kern-Minimum bei einer einzigen VMware-Lizenz kosten würde

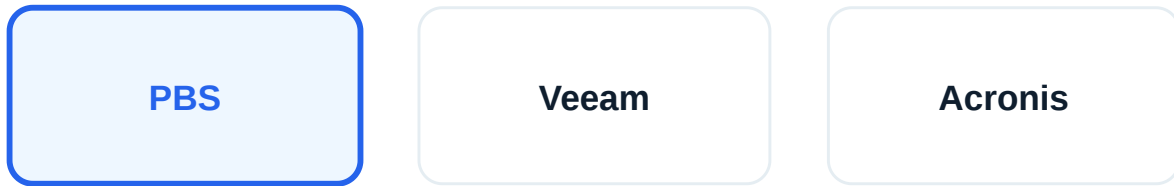
Entscheidungsleitfaden



- Klein bis mittel, kostenbewusst → Proxmox, insbesondere wenn ein Dienstleister die Support-Rolle übernimmt
- Großes VMware-Ökosystem mit Spezialanforderungen nötig → VMware bleibt im Rennen, trotz der gestiegenen Kosten unter Broadcom
- Reine Windows-Welt ohne Linux-Bedarf → Hyper-V mitprüfen, besonders wenn ohnehin überall Windows-Server-Lizenzen vorhanden sind
- Entscheidend ist nicht Ideologie, sondern die ehrliche TCO-Rechnung aus dem vorigen Block – Passform zur eigenen Umgebung schlägt Bauchgefühl

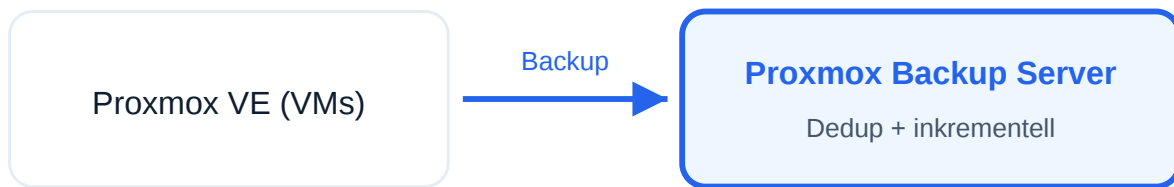
M13 · Backup

Eigene Lösung (PBS) + Veeam + Acronis – Vergleich und 3-2-1-Strategie



Alle drei im Dienste der 3-2-1-Strategie

Proxmox Backup Server (PBS)

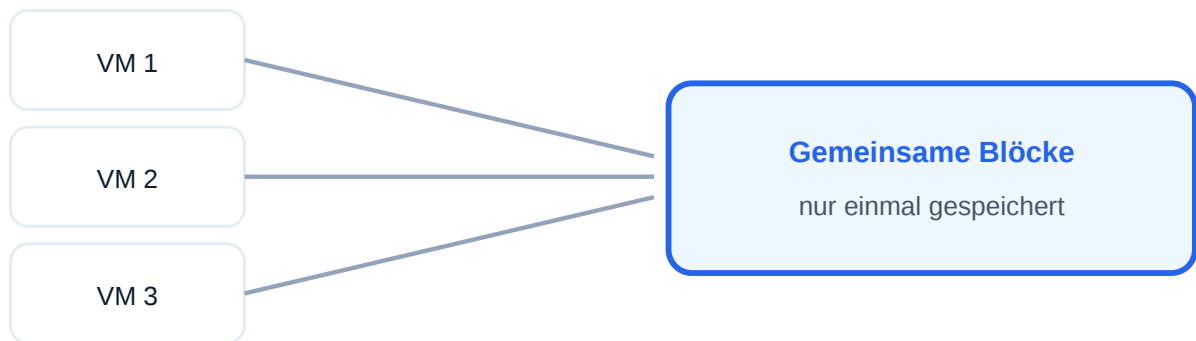


Schnelle, granulare Restores – ganze VM oder einzelne Dateien

PBS ist eine eigenständige Software – installierbar auf dedizierter Hardware, als VM oder Container, per eigenem Debian-basiertem ISO oder als Paket auf bestehendem Debian. Kommunikation mit Proxmox VE läuft über eine eigene API und den Client `proxmox-backup-client`.

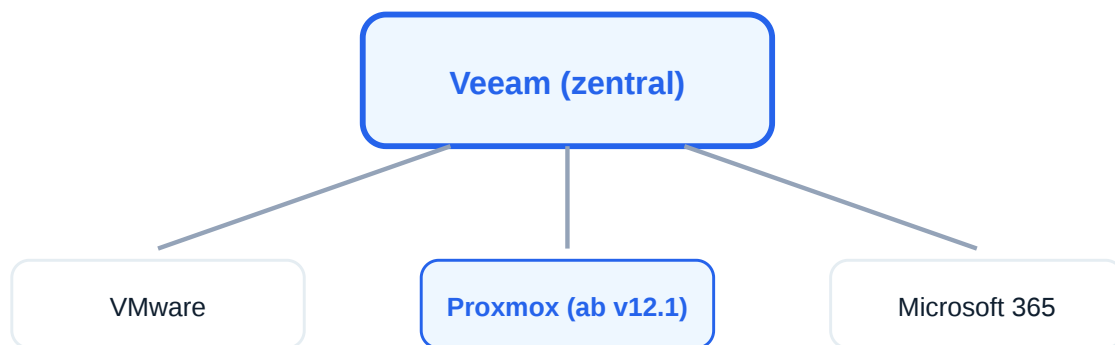
- Eigene, kostenlose Backup-Lösung von Proxmox – keine Lizenzkosten, Enterprise-Support ist optional als kostenpflichtiges Abo erhältlich
- Deduplizierung und inkrementelle Backups auf Basis von **Content-Defined Chunking**: Daten werden in variabel große Blöcke (im Schnitt ca. 4 MiB) zerlegt und über SHA-256-Prüfsummen wiedererkannt
- Schnelle, granulare Restores – ganze VM, einzelne Festplatte oder einzelne Datei aus dem Dateisystem-Backup-Format `pxar` heraus, direkt im PBS-Webinterface durchsuchbar
- Sichert nicht nur VMs und Container, sondern per Agent auch physische Server sowie ganze Proxmox-VE- und PBS-Hosts selbst
- Aufbewahrung über Regeln wie `keep-daily`, `keep-weekly` und `keep-monthly` (Grandfather-Father-Son-Prinzip), automatisch per Prune-Job durchgesetzt

Wie PBS spart



- Nach dem ersten vollen Backup überträgt PBS nur noch geänderte Blöcke – bei QEMU-VMs erkannt über **Dirty-Bitmaps**, die der Hypervisor seit dem letzten Backup mitführt
- Deduplizierung wirkt global über alle VMs, alle Nodes und alle Backup-Läufe hinweg im selben Datastore – ein Block, der in zehn ähnlichen Windows-VMs vorkommt, wird nur einmal gespeichert
- Praxisbeispiel: Zehn nahezu identische Windows-Server-VMs mit je 40 GB belegtem Speicher benötigen ohne Dedup rechnerisch 400 GB, mit PBS-Dedup je nach Ähnlichkeit oft nur einen Bruchteil davon
- Verify-Jobs prüfen periodisch die Prüfsummen aller gespeicherten Chunks und decken so stille Datenfehler (Bit Rot) auf, bevor sie beim Restore auffallen
- Garbage Collection räumt nicht mehr referenzierte Chunks auf, sobald alte Backups per Prune-Regel gelöscht wurden – Speicher wird also erst nach dem nächsten GC-Lauf wirklich frei
- Ergebnis: wenig Speicherbedarf, kurze Backup-Fenster – typische inkrementelle Läufe dauern oft nur Minuten statt Stunden

Veeam



- Nativer Proxmox-Support seit **Veeam Backup & Replication 12.1** (Anfang 2024) – vorher nur über Umwege möglich, ein klares Signal, dass der Marktführer Proxmox ernst nimmt
- Bekannt und etabliert aus der VMware-Welt, viele Admins kennen die Konsole bereits aus jahrelangem Einsatz
- Ein Werkzeug für die ganze gemischte Landschaft: VMware, Hyper-V, physische Server, Cloud-Workloads und Microsoft 365 laufen in derselben Konsole wie Proxmox
- Application-aware Processing: konsistente Sicherung von Datenbanken und Exchange/SQL-Server inklusive Log-Truncation direkt aus dem Backup-Job heraus
- Unterstützt unveränderliche (immutable) Backup-Repositories – wichtiger Ransomware-Schutz, den PBS über eigene Datastores und Sync-Jobs anders, aber vergleichbar löst
- Kostenpflichtig, Lizenzierung meist pro Socket oder pro geschützter Instanz – für reine Proxmox-Neuumbungen ohne bestehende Veeam-Basis selten die erste Wahl

Acronis



Agent pro VM, statt zentral am Hypervisor

Cyber-Protection-Ansatz, weniger tief in Proxmox integriert

- Agent-basiert – die Backup-Software wird einzeln in jeder VM installiert statt zentral am Hypervisor anzusetzen; das bedeutet mehr Verwaltungsaufwand pro Maschine, ermöglicht aber applikationsnahe Sicherung
- **Acronis Cyber Protect** vereint Backup mit Antivirus, Anti-Ransomware und Patch-Management in einer Konsole – ein „Cyber Protection“-Ansatz statt reinem Backup-Werkzeug
- Weniger tief in Proxmox integriert als PBS oder natives Veeam – Acronis ist plattformübergreifend gedacht, nicht Proxmox-spezialisiert
- Lizenzierung meist pro geschütztem Workload (Agent), nicht pro Host – Kosten skalieren mit der Anzahl der VMs, nicht mit der Cluster-Größe
- Für IT-Dienstleister, die Endgeräte und Kunden-Clients bereits über **Acronis Cyber Protect Cloud** verwalten, ist die Erweiterung auf Proxmox-VMs oft der pragmatische Weg, um kein weiteres separates Werkzeug einzuführen
- Als Neuanschaffung ausschließlich für Proxmox selten die erste Wahl – als bereits vorhandenes Werkzeug, das man mitnutzt, durchaus sinnvoll

Vergleich PBS / Veeam / Acronis

LÖSUNG	KOSTEN · INTEGRATION · RESTORE
PBS	kostenlos · tief · schnell/granular
Veeam	kostenpflichtig · nativ tief · schnell/granular
Acronis	kostenpflichtig · breit statt tief · agent-abhängig
Passung je nach Umgebung: rein Proxmox → PBS, gemischt → Veeam	

PBS

Kostenlos, tiefste Integration, schnellste granulare Restores – Standardwahl für reine Proxmox-Umgebungen

Veeam

Kostenpflichtig, nativer Proxmox-Support seit 12.1, stark bei gemischten Landschaften mit vorhandener Veeam-Basis

Acronis

Kostenpflichtig, agent-basiert, Cyber-Protection-Ansatz – sinnvoll, wenn bereits für Endgeräte im Einsatz

- Kosten: PBS kostenlos (Enterprise-Support optional), Veeam und Acronis lizenzpflichtig, meist pro Socket bzw. pro Workload
- Integrationstiefe und Restore-Tempo: PBS und natives Veeam liegen vorn, beide granular bis auf Dateiebene
- Bare-Metal- und plattformübergreifende Sicherung: Domäne von Veeam und Acronis, PBS ist auf die Proxmox-Welt fokussiert
- Passung richtet sich nach der jeweiligen Umgebung, nicht nach Vorliebe: reines Proxmox-KMU → PBS, gemischte oder größere Landschaft mit vorhandenem Veeam → Veeam, Acronis bereits im Einsatz → mitnutzen

3-2-1 & Konsistenz

3 Kopien

2 Medien

1 offsite

Konsistenz per Guest-Agent-Freeze, Restore regelmäßig testen

- 3 Kopien der Daten, auf 2 verschiedenen Medien, davon 1 an einem anderen Ort – die Erweiterung „3-2-1-1-0“ ergänzt eine unveränderliche/Offline-Kopie und null Fehler nach Verifizierung
- Backup-Konsistenz sicherstellen: der `qemu-guest-agent` in der VM friert das Dateisystem kurz ein (Windows über VSS, Linux über `fsfreeze`), damit kein halb geschriebener Zustand gesichert wird
- Offsite-Kopie in der Praxis: ein zweiter PBS an einem anderen Standort, angebunden per Sync-Job, der neue Chunks automatisch repliziert
- Retention nach GFS-Prinzip planen, z. B. tägliche Backups 7 Tage, wöchentliche 4 Wochen, monatliche 12 Monate vorhalten
- Restore regelmäßig testen – ein ungetestetes Backup ist nur eine Hoffnung, der Ernstfall ist der falsche Moment, um zu merken, dass es nicht startet
- DSGVO-Bezug (siehe Modul 4): Retention-Regeln nicht nur nach Recovery-Bedarf, sondern auch nach dem Löschkonzept planen – wird ein Datensatz im Produktivsystem gelöscht, sollte er nicht unbegrenzt lange in alten Backups fortleben; die GFS-Fristen liefern damit gleichzeitig eine sinnvolle Obergrenze für beides

M14 · Migrations-Praxis (Abschluss)

Alles zur Checkliste gebündelt – VMware → PVE und Hyper-V → PVE



Checkliste VMware → Proxmox

- ✓ VMware Tools raus, VirtIO rein
- ✓ Import-Assistent (ESXi 8.2+) oder OVF
- ✓ Boot-Modus, Netz, Aktivierung prüfen

Proxmox VE bringt seit Version 8.2 einen eigenen **Import-Assistenten** mit, der sich direkt mit einem ESXi-Host oder vCenter verbindet und VMs samt Konfiguration übernimmt – ohne Umweg über manuellen OVF-Export/Import.

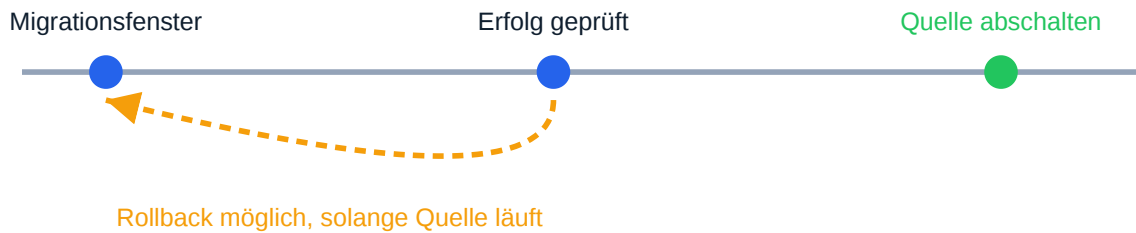
- **VMware Tools** vor der Migration sauber deinstallieren (über Programme/Features oder `vmware-toolbox-cmd`) – sonst können alte Treiber nach dem Wechsel Konflikte verursachen
- **VirtIO-Treiber** vorbereiten: das `virtio-win`-Treiber-ISO einhängen und insbesondere den Storage-Treiber `viostor` vor dem ersten Boot unter Proxmox installieren – sonst startet Windows mit Bluescreen 0x7B (kein Bootdatenträger gefunden)
- Übernahme entweder per Import-Assistent direkt vom ESXi/vCenter (ab PVE 8.2) oder klassisch per OVF-Export mit `ovftool` und anschließend `qm importovf`
- Alternativ Festplatten einzeln konvertieren: `qemu-img convert -f vmdk quelle.vmdk -O qcow2 ziel.qcow2`, danach per `qm importdisk` der Ziel-VM zuordnen
- Boot-Modus prüfen: BIOS-VMs bleiben bei SeaBIOS, EFI-VMs benötigen in Proxmox den OVMF-Bios-Typ mit passender EFI-Disk – eine Fehlkonfiguration hier verhindert den Boot komplett
- Netzwerk: vSwitch-Portgruppen auf Linux-Bridges (`vmbr0` etc.) abbilden und IP-Konfiguration prüfen – bei DHCP kann sich durch die neue virtuelle MAC-Adresse die zugewiesene IP ändern
- Mit Windows-Neuaktivierung rechnen: der Wechsel der virtuellen Hardware ändert den Hardware-Fingerprint, KMS- oder MAK-Aktivierung meldet sich unter Umständen neu

Checkliste Hyper-V → Proxmox

- ✓ Integrationsdienste raus, VirtIO rein
- ✓ VHDX → qcow2/raw (qemu-img)
- ! Gen-2 = UEFI, TPM für Windows 11

- Hyper-V-Integrationsdienste in der VM deinstallieren (Systemsteuerung → Programme oder VM-Einstellungen), danach VirtIO-Treiber-ISO einhängen und installieren
- Wichtiger Stolperstein vorab: vorhandene Hyper-V-**Checkpoints** (frühere „Snapshots“) in Hyper-V-Manager zusammenführen bzw. löschen, bevor exportiert wird – sonst bricht die Snapshot-Kette beim Export ab
- Festplatten konvertieren: `qemu-img convert -f vhdx quelle.vhdx -O qcow2 ziel.qcow2` wandelt direkt, ohne Umweg über das ältere VHD-Format – vorher mit `qemu-img info quelle.vhdx` prüfen, ob eine dynamische oder fixe Disk vorliegt
- Generation-1-VMs sind BIOS/MBR-basiert und lassen sich meist direkt mit SeaBIOS starten; **Generation-2-VMs** sind UEFI mit GPT-Partitionstabelle und benötigen in Proxmox den OVMF-Bios-Typ
- Für Windows 11 unter Generation 2 zusätzlich ein virtuelles TPM einrichten: `qm set VMID --tpmstate0 storage:1,version=v2.0`, da Windows 11 zwingend TPM 2.0 voraussetzt
- Nach dem Boot Netzwerkadapter neu einrichten (Hyper-V nutzt eigene Adaptertypen) und Windows-Aktivierung wie bei VMware-Migrationen prüfen

Downtime-Fenster & Rollback



Migrationsfenster realistisch kalkulieren statt Wunschdenken kommunizieren – lieber ein großzügiges Fenster ankündigen und früher fertig sein als umgekehrt.

- Faustregel für die Datenübertragung: bei 1-GbE-Anbindung realistisch ca. 100 MB/s, ein 1-TB-Disk-Image braucht damit grob 3 Stunden reine Kopierzeit – Grundlage für die Fenster-Kalkulation
- „Seed & Sync“-Ansatz für große VMs: den Großteil der Daten Tage vorher in Ruhe kopieren, im eigentlichen Wartungsfenster nur noch das Delta seit dem letzten Sync übertragen und dann umschalten – verkürzt die eigentliche Downtime auf einen Bruchteil
- Goldene Regel: die alte Umgebung erst abschalten, wenn die neue nachweislich läuft – nie beide Brücken gleichzeitig abbrechen
- Rollback-Fenster einplanen: Quelle (VMware-/Hyper-V-VM) mindestens 1–2 Wochen ausgeschaltet, aber nicht gelöscht bereithalten, bevor sie endgültig abgebaut wird
- Sofort nach der Migration einen Proxmox-Snapshot der neuen VM anlegen – geht in den ersten Tagen etwas schief, kann auch Proxmox-seitig zurückgerollt werden, ohne auf die alte Quelle zurückgreifen zu müssen
- DNS-TTL vor dem Umschalten absenken und IP-Adressen nach Möglichkeit beibehalten, um Folgeänderungen an Firewalls und externen Verweisen zu vermeiden

Kapazitätsplanung / Sizing



Nicht knapp auf Kante planen

- Reale Auslastung statt zugewiesener Ressourcen erfassen: CPU-, RAM- und Storage-Nutzung über 2–4 Wochen aus vCenter- bzw. Hyper-V-Performance-Daten auswerten, nicht nur die konfigurierte Größe übernehmen
- Realistische Überbuchung einplanen: vCPUs typischerweise im Verhältnis 2:1 bis 4:1 zu physischen Kernen möglich, RAM dagegen eher konservativ dimensionieren (Ballooning existiert, ist bei performancekritischen Workloads aber riskant)
- Reserve für Wachstum und für den HA-Ausfallfall einplanen: fällt in einem 4-Node-Cluster ein Node aus, müssen die verbleibenden drei dessen VMs mittragen können
- Faustregel: Cluster im Normalbetrieb nicht über ca. 70–80 % Auslastung pro Node fahren, sonst fehlt nach einem Node-Ausfall der Platz, damit betroffene VMs per HA neu starten
- Storage-Thin-Provisioning nur mit Monitoring einsetzen – tatsächliche Belegung regelmäßig prüfen, damit der Pool nicht unbemerkt vollläuft
- Nicht auf Kante planen: Reserve ist keine Verschwendung, sondern die Versicherung gegen den Ausfalltag

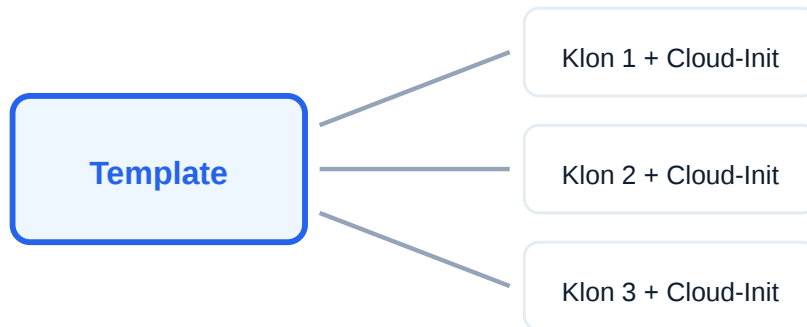
Automatisierung



Ersatz für PowerCLI – wiederholbare Deployments

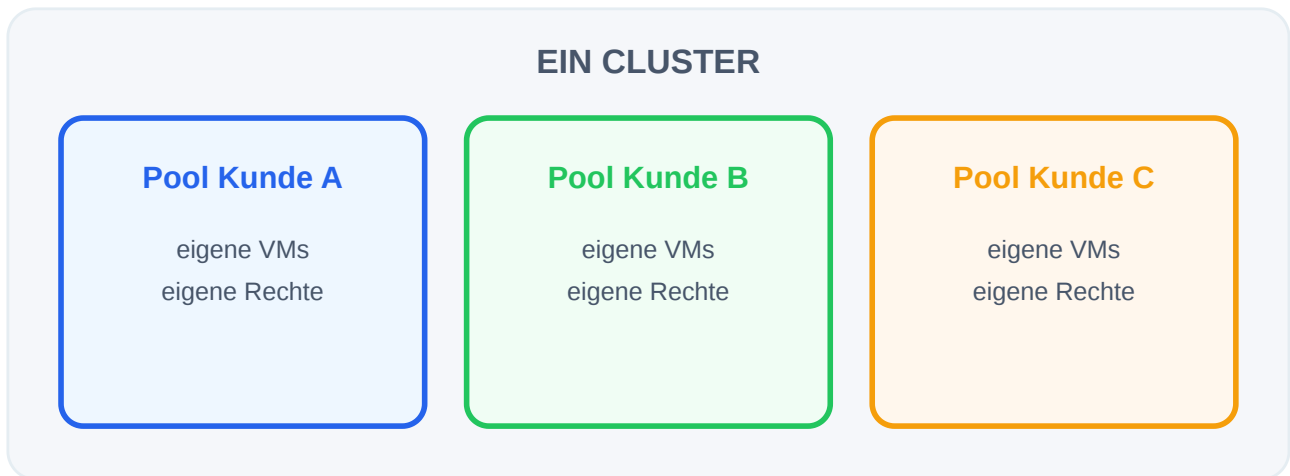
- Proxmox bietet eine vollständige REST-API (Port 8006) mit API-Token-Authentifizierung als Ersatz für PowerCLI-Skripte – jede Aktion der Web-UI lässt sich auch per API auslösen
- **Ansible** für Konfiguration und Rollout, meist über Community-Collections mit Proxmox-Modulen; **Terraform** (z. B. Provider `bpg/proxmox`) für das deklarative Aufsetzen ganzer Umgebungen
- Wiederholbare Deployments statt Klickarbeit: einmal beschrieben, beliebig oft identisch ausgerollt
- Für den Dienstleister mit vielen Kunden besonders lohnend: eine saubere Vorlage einmal bauen und pro Kunde nur die Parameter (IPs, Ressourcengrößen, Hostnamen) austauschen
- Infrastructure-as-Code-Definitionen versioniert in Git ablegen – nachvollziehbare Änderungshistorie, einfacheres Onboarding neuer Kollegen und Basis für Compliance-Nachweise
- Für einzelne Kunden mit wenigen VMs lohnt sich der Aufwand oft nicht, für wiederkehrende Standard-Rollouts über viele Kunden hinweg dagegen deutlich

Templates & Cloud-Init



- Template-Erstellung: VM einmal sauber aufsetzen, benötigte Pakete/Agenten (u. a. `qemu-guest-agent`) installieren, dann mit `qm template VMID` in ein Template umwandeln
- Bei Windows-Vorlagen vorher **Sysprep** ausführen, damit jeder Klon eine eigene SID erhält – ohne Sysprep teilen sich alle Klone dieselbe SID, was vor allem in Active-Directory-Umgebungen zu Problemen führt
- Bei Linux-Vorlagen Maschinen-ID und SSH-Host-Keys vor der Template-Umwandlung entfernen (z. B. `truncate -s0 /etc/machine-id`), damit jeder Klon beim ersten Boot frische Werte generiert
- Linked Clones (`qm clone TMPLID NEWID --full 0`) teilen sich die Basis-Disk des Templates und sparen dadurch Speicher und Klonzeit gegenüber Full Clones
- Cloud-Init über eine eigene virtuelle Disk am Template: konfiguriert Hostname, Netzwerk, Benutzer und SSH-Keys beim ersten Start automatisch, ganz ohne Login – funktioniert am besten mit vorbereiteten Cloud-Images (z. B. offizielle Debian/Ubuntu-Cloud-Images)
- Ergebnis: standardisierte, sofort einsatzbereite VMs in Sekunden bis Minuten statt manueller Ersteinrichtung pro Kunde

Mandantenfähigkeit (MSP)



- Ressourcen-Pools gruppieren die VMs, Storage und weitere Objekte eines Kunden zu einer Einheit innerhalb desselben Clusters
- Rechte und ACLs pro Pool vergeben, z. B. über Rollen wie `PVEVMAdmin` oder `PVEAuditor` – Kunde A sieht und verwaltet nur seine eigenen Maschinen
- Anbindung an AD/LDAP oder eine separate Proxmox-Realm pro Kunde ermöglicht granulare, delegierte Administration ohne gemeinsame Zugangsdaten
- Wichtige Einschränkung: Proxmox erzwingt über Pools keine harten Ressourcen-Kontingente wie z. B. VMware vCloud Director – Kapazitätsplanung und Monitoring müssen das im Blick behalten
- Pools trennen nur die Verwaltung, nicht das Netzwerk: für echte Mandantentrennung zusätzlich VLANs (802.1Q) oder Proxmox SDN mit eigenen Zonen/VNets pro Kunde einsetzen
- Erst Pools plus Rechte und VLANs/SDN zusammen ergeben echte Mandantentrennung – eines allein ist nur die halbe Miete

Lessons Learned

1. Vorbereitung schlägt Improvisation

2. Immer testen: Netz, Aktivierung, Restore

3. Testlabor vor Produktion

Panik ist ein schlechter Architekt, ein Testlabor ist ein guter Lehrer

Panik ist ein schlechter Architekt, aber ein Testlabor ist ein guter Lehrer – wer Proxmox in Ruhe geübt hat, migriert gelassen.

- Vorbereitung schlägt Improvisation – immer: eine dokumentierte Migration (Netzplan, IP-Liste, Zugangsdaten, Lizenzschlüssel) läuft entspannter und schneller ab als eine spontane
- Immer testen: Netzwerkerreichbarkeit, Windows-/Software-Aktivierung, Anwendungsfunktion – und vor allem den ersten Restore aus dem neuen PBS-Backup, bevor die Migration als abgeschlossen gilt
- Erst ins Testlabor, dann auf den Kundenserver – ein Testcluster (notfalls auch nested auf einer einzelnen kräftigen Maschine oder auf ausgemusterter Hardware) deckt Überraschungen auf, bevor sie beim Kunden passieren
- Komplexe Fälle wie Domain-Controller oder Datenbank-Cluster vorher im Testlabor Schritt für Schritt durchspielen – das macht die Zeitschätzung fürs echte Wartungsfenster belastbar statt geraten