Proxmox Home Lab Guide 2024

Author: Brandon Lee
https://www.virtualizationhowto.com

This Proxmox home lab e-book contains a compilation of posts I have created in working with Proxmox in the home lab. While not an exhaustive guide to Proxmox, the posts compiled in this guide cover the basic installation and configuration of Proxmox, including networking, storage, security, monitoring, and other topics. I have also included a section on installing pfSense on Proxmox and other specialized lab scenarios.

Since many will be coming from a VMware vSphere background, the guide starts from installing Proxmox in a nested environment running on VMware vSphere. We conclude coming full circle by installing nested ESXi inside of Proxmox.

Table of Contents

Nested Proxmox VMware installation in ESXi

January 13, 2022
Proxmox

Booting the Proxmos 7.1 VE installer

In working with clients and different environments, you will definitely see many different hypervisors used across the landscape of enterprise organizations. While I recommend VMware vSphere for business-critical enterprise workloads to customers, there are use cases where I see other hypervisors used. Proxmox is a really great open-source, free hypervisor available for use and is even developed for use in enterprise applications. I also know of many in the community running Proxmox in their home lab environment. If you are like me and like to play around with technology, hypervisors, and other

cool geeky stuff, I find I load a lot of different solutions in the lab. Let's take a look at nested Proxmox VMware installation in ESXi and see how you can easily spin up a Proxmox host in a vSphere VM.

What is Proxmox?

Proxmox is easily administered using a rich, fully-featured web interface that actually looks and feels nice. While it is not in my opinion where the vSphere client is in look and feel, it is quite nice and does the job needed to administer the Proxmox environment.

Proxmox VE is an open-source hypervisor platform for enterprise virtualization. It provides many features needed to run production workloads, including virtual machines, containers, software-defined storage, networking, clustering, and other capabilities out-of-the-box. It is based on Linux, so you get the pure Linux experience for virtualization, containers, and other facets. Note some of the benefits:

- Open-source software
- No vendor lock-in
- Linux kernel
- Fast and easy installation
- Easy-to-use with the intuitive web-based management interface
- Low administration costs and simple deployment
- Huge active community

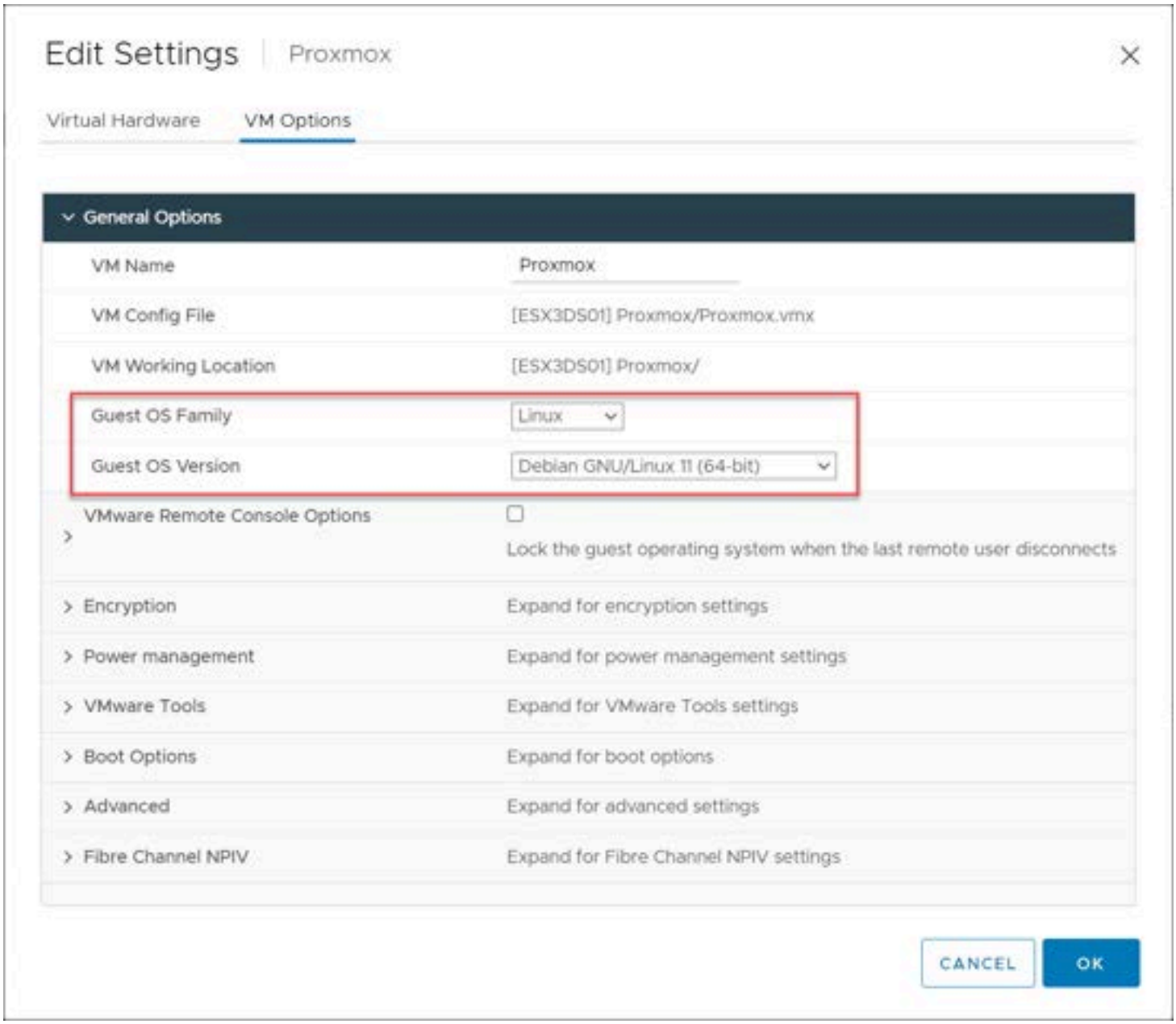
Nested Proxmox VMware installation in ESXi

The first thing you need for your nested Proxmox VMware installation in ESXi is to download the Proxmox ISO for installation. You can download the Proxmox ISO here:

- Get the free Proxmox VE ISO installer
- Current version Proxmox VE 7.1

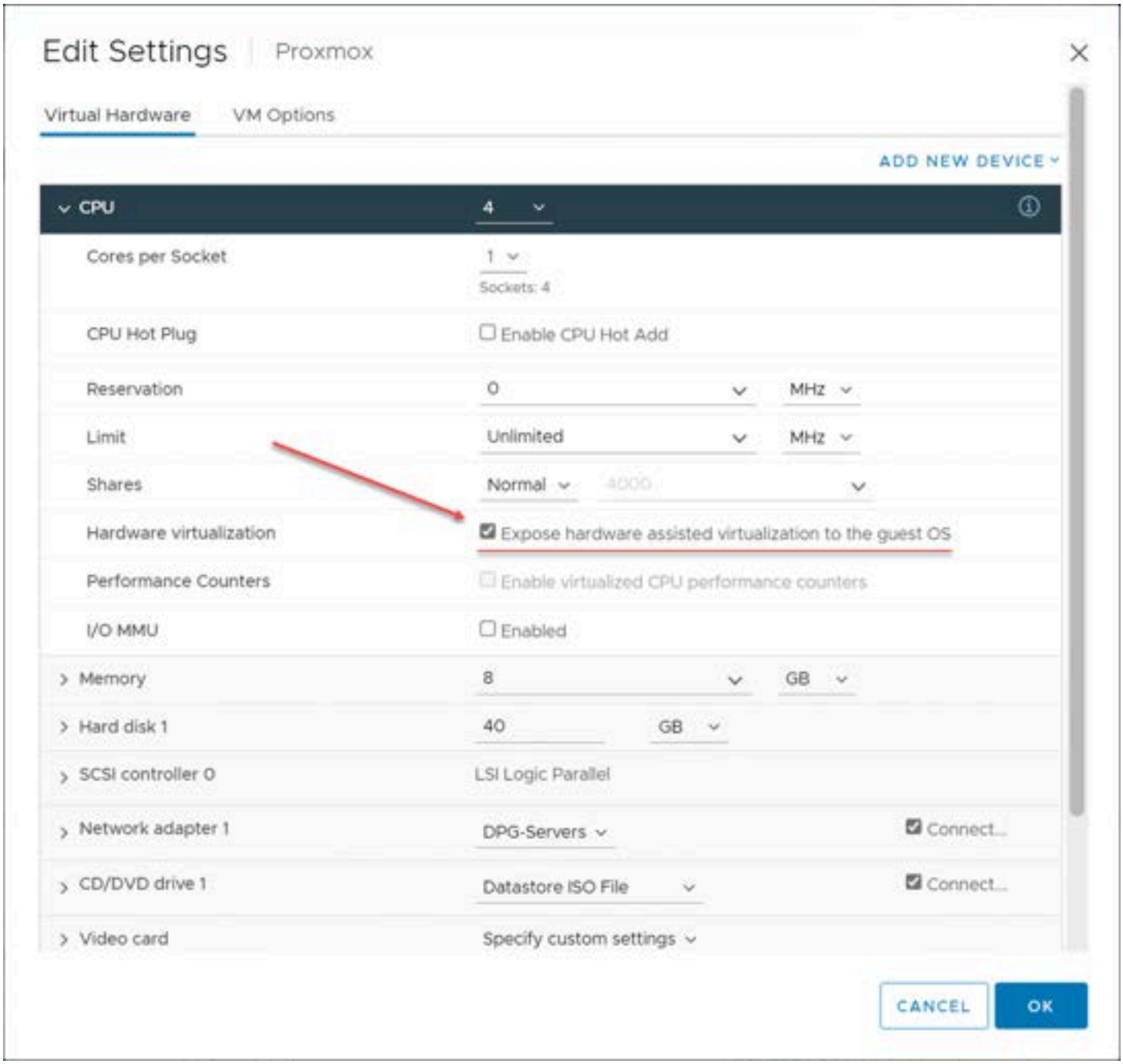
You will mount the ISO to your virtual machine in VMware vSphere like you would any other OS installation. Create a new VMware vSphere virtual machine with the following details:

- Guest OS Family – **Linux**
- Guest OS Version – **Debian GNU/Linux 11 (64-bit)**



Proxmox VMware virtual machine settings

Next, make sure to expose hardware-assisted virtualization to the guest OS for your soon-to-be Proxmox installation. As most of us are familiar with in our nested ESXi labs, this is a simple checkbox in the properties of your VMware ESXi virtual machine under the CPU settings.

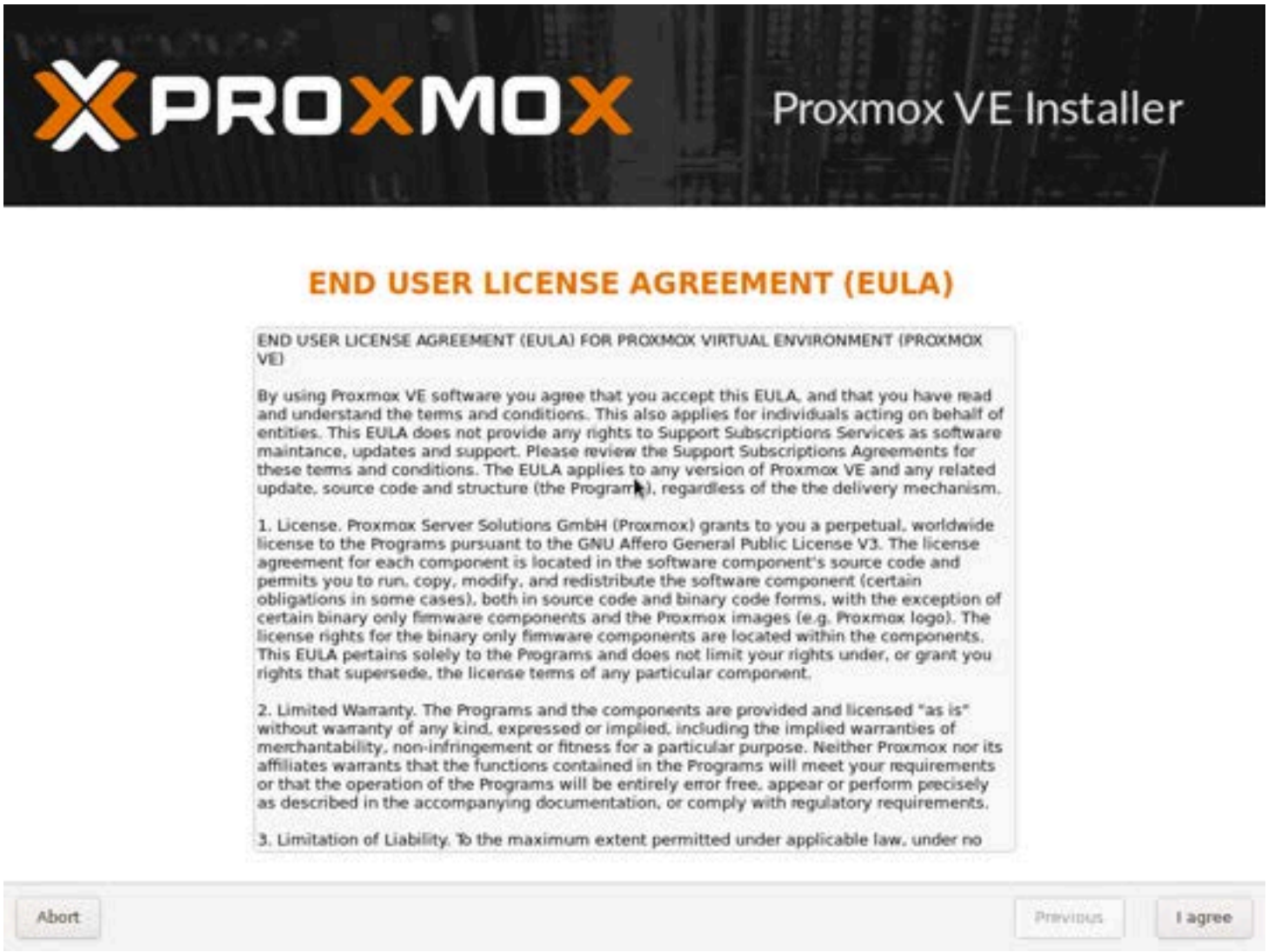


Exposing CPU hardware virtualization to the guest OS

After booting from the ISO, the Proxmox VE 7.1 installation begins. Select to **Install Proxmox VE**.

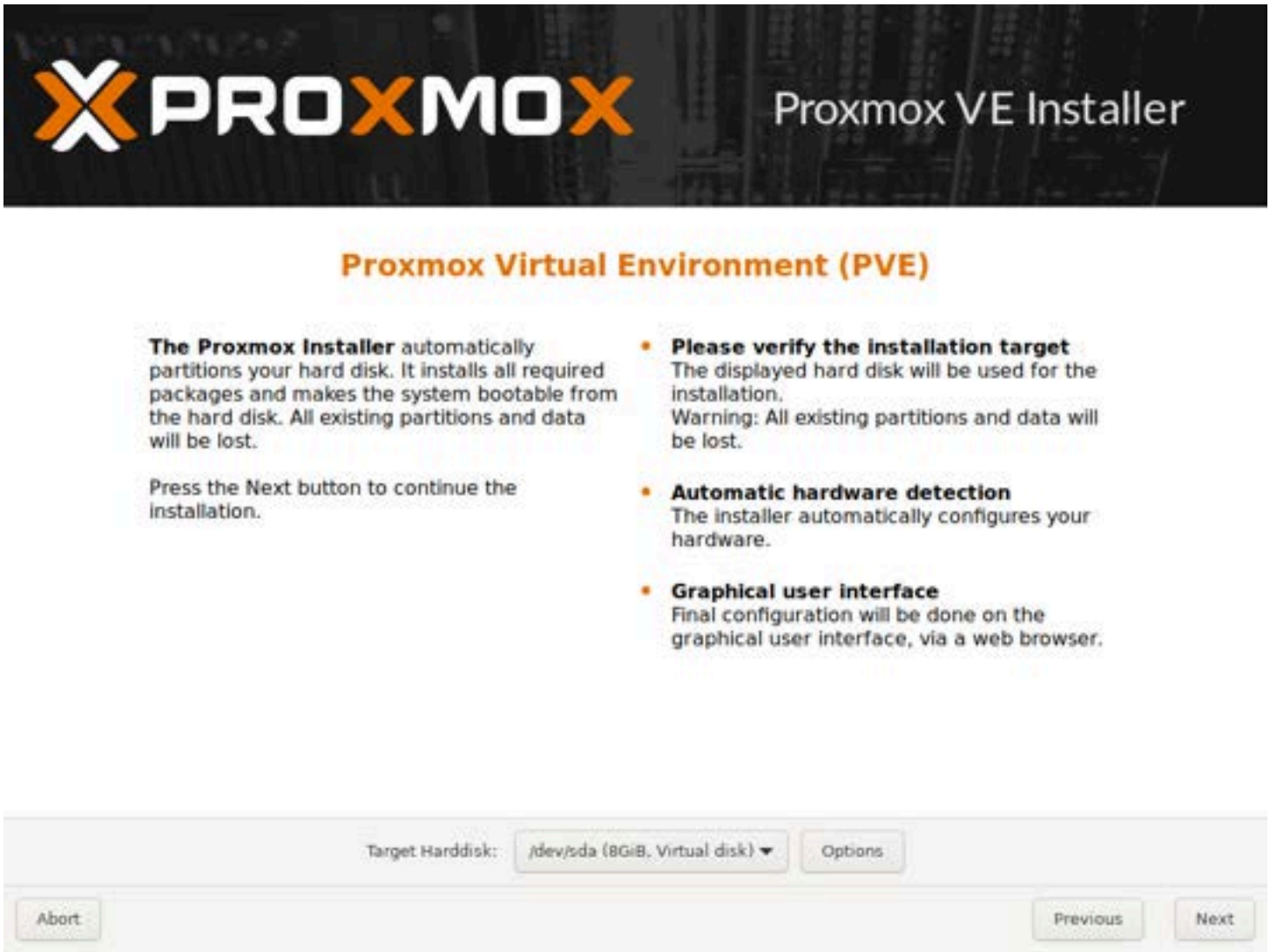
Booting the Proxmos 7.1 VE installer

First things first. Accept the EULA to proceed.



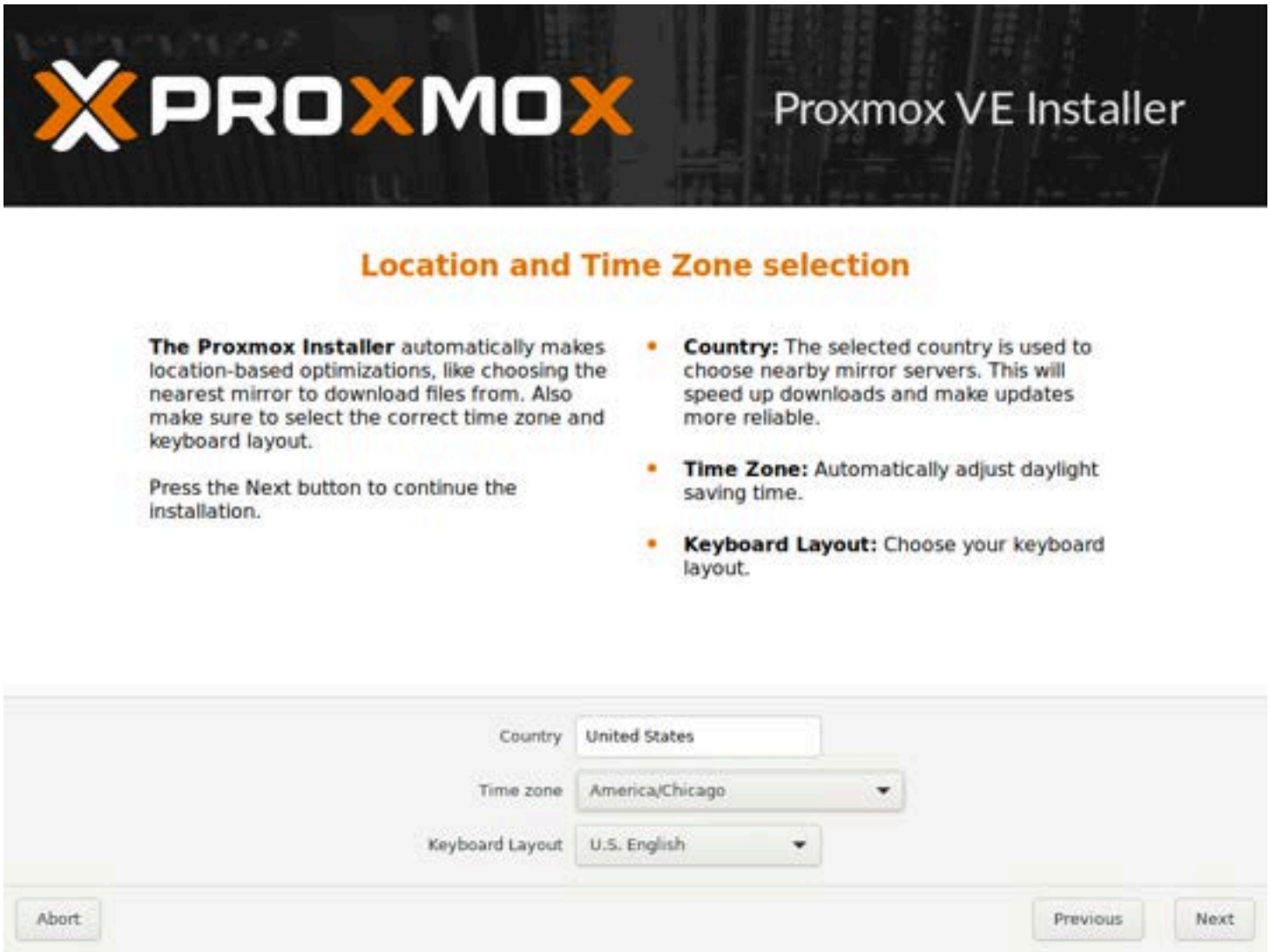
Accept the EULA for Proxmox VE 7.1

Next, you can customize the disk partition layout if you choose. However, for my nested Proxmox VMware installation, I am accepting the defaults.



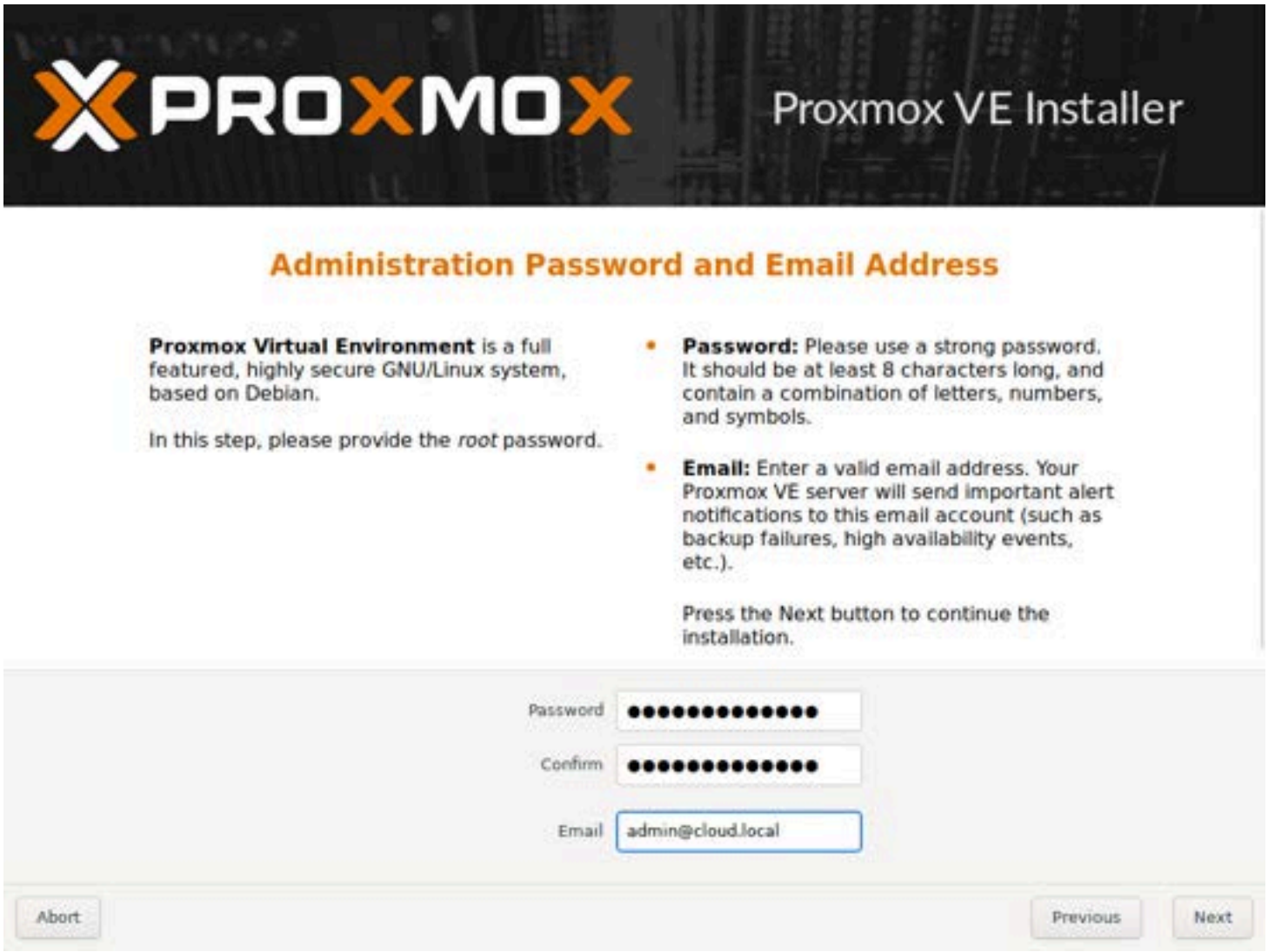
Select the disk partitioning to be used with the Proxmox VE 7.1 installation

Next up is setting your location and time zone configuration.



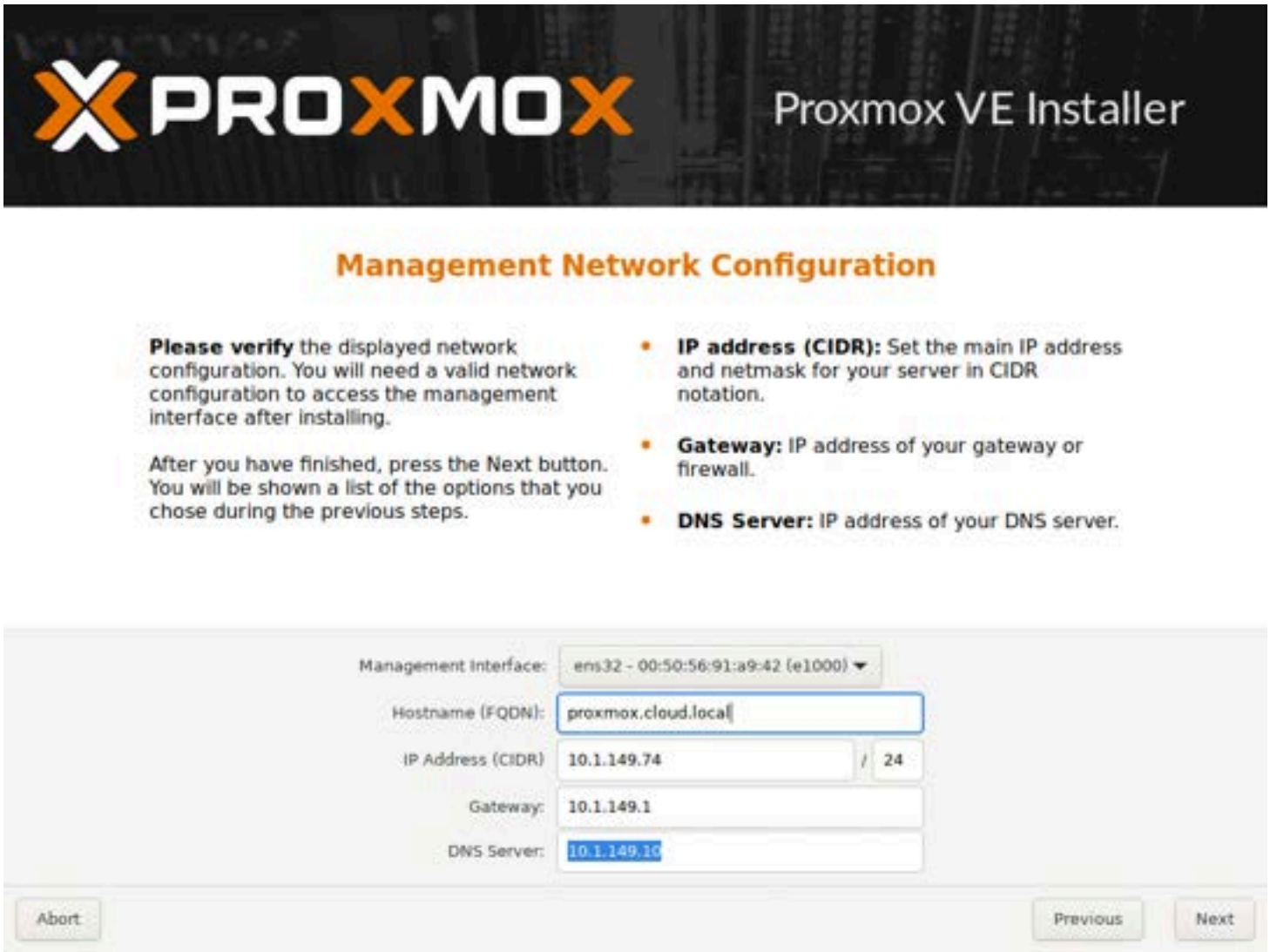
Set the location and time zone

Configure the password for your **root** account. Also, Proxmox has you enter an email address.



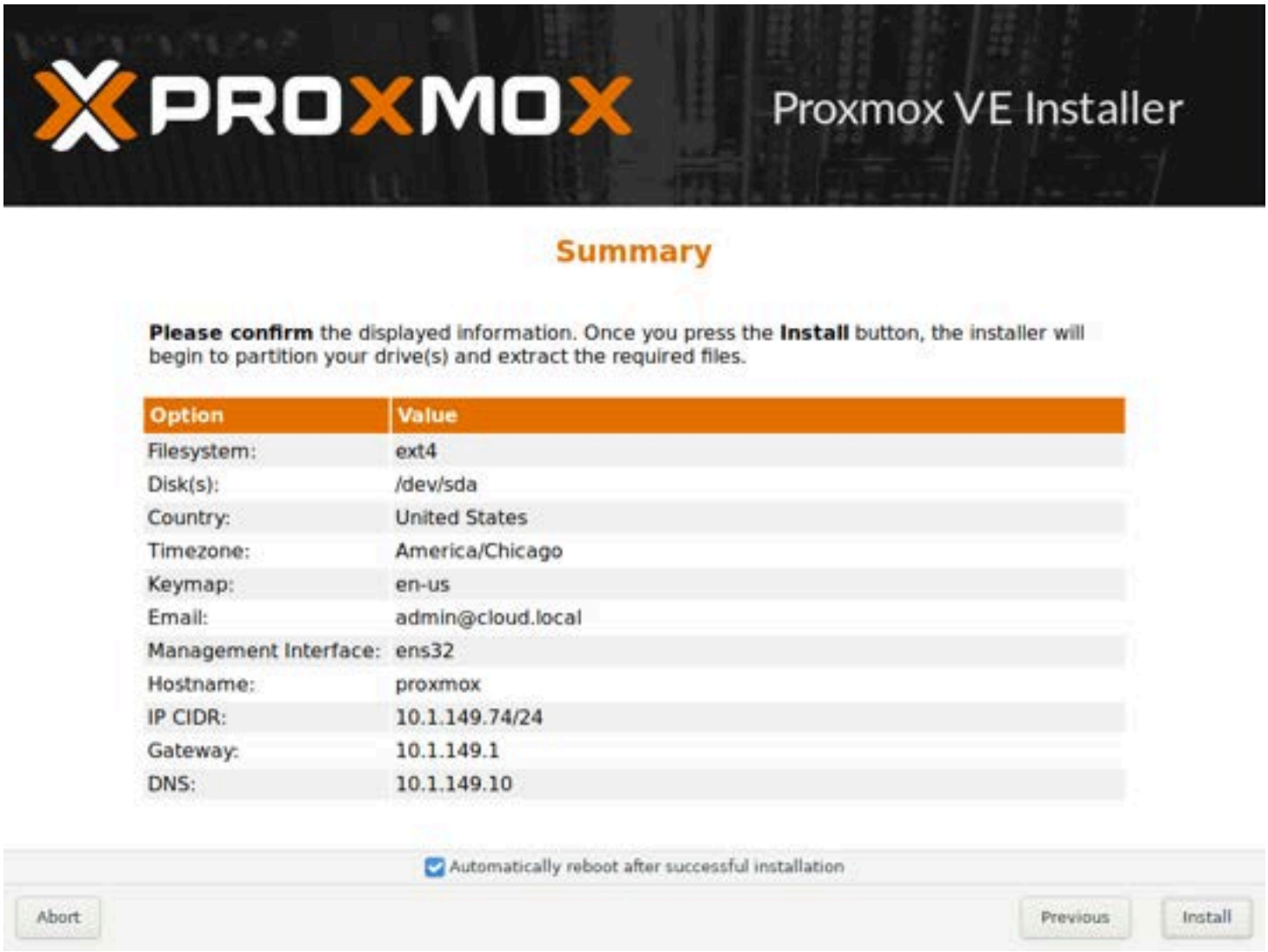
Set the administrator password and email address

Configure the Proxmox hostname and your network configuration.



Set the hostname and network configuration

Finally, we come to the Summary screen. Here, review the configuration and validate your settings. Then, click **Install**.

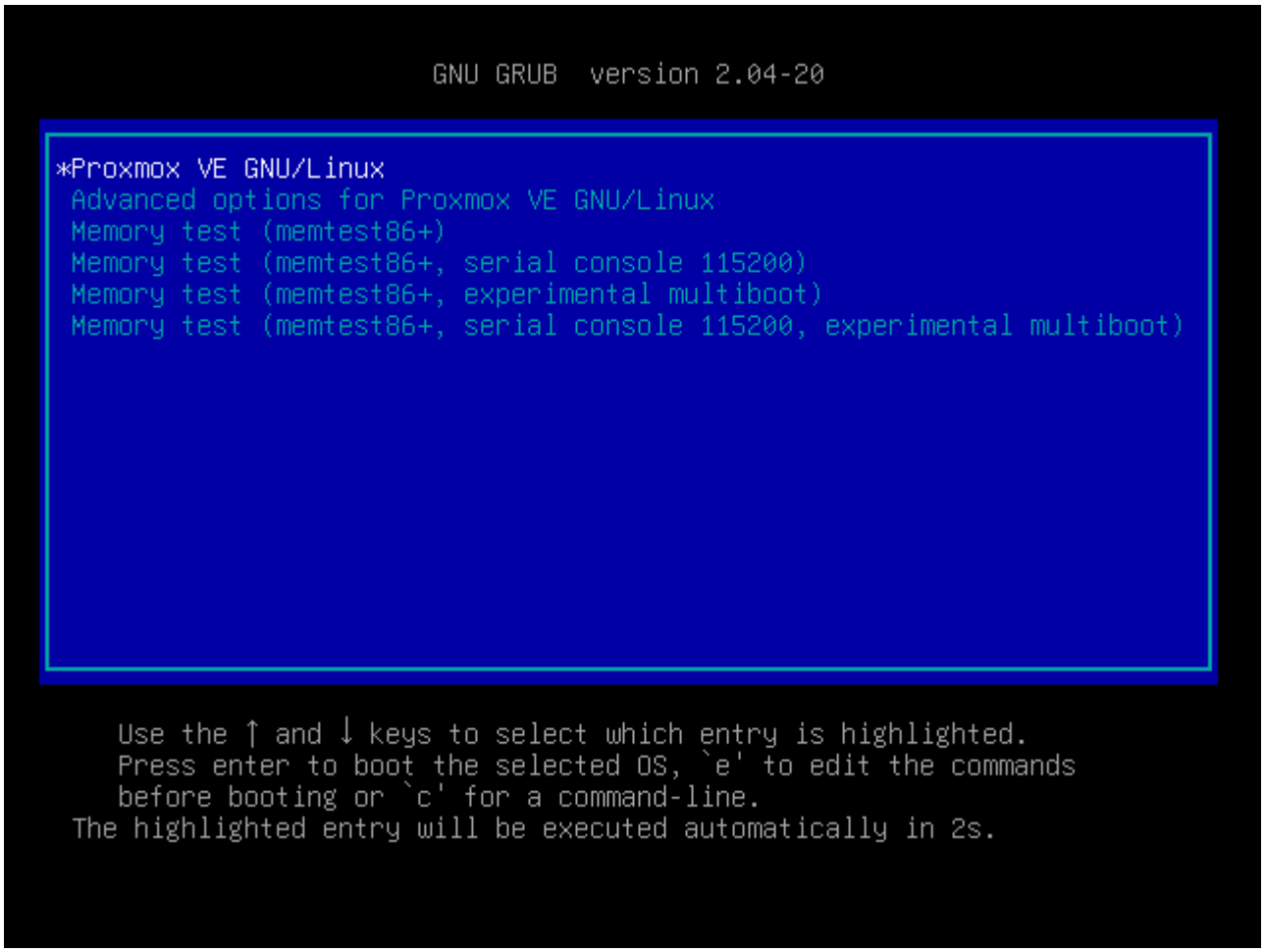


Summary of the Proxmox VE 7.1 installation

The installation process begins.

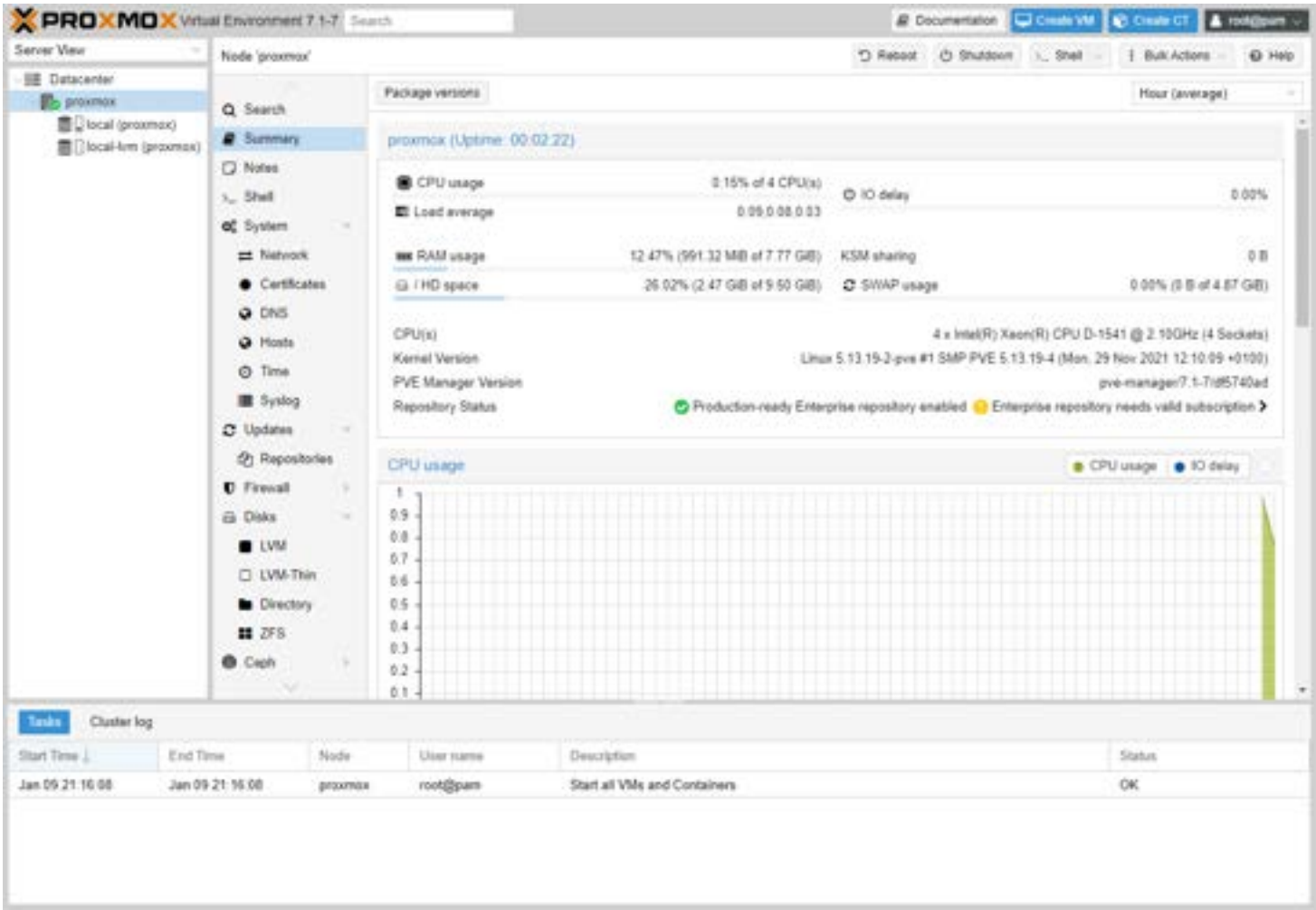
Proxmox VE 7.1 installation proceeds

After finishing the installation, the Proxmox server will reboot. Below is the boot screen captured as it reboots from the installation.



Proxmox VE 7.1 boots as a VMware ESXi VM

Finally, we are logged into the Proxmox web GUI using root and the password configured during the installation. Overall, the nested Proxmox VMware installation in ESXi was straightforward and easy. If you want to play around with Proxmox in a nested configuration, VMware vSphere provides a great way to do this using the basic functionality we have used for quite some time with nested ESXi installations.



Logged into the Proxmox VE 7.1 web interface

Wrapping Up

Proxmox is a cool hypervisor that provides a lot of features in an open-source, freely available download. The latest Proxmox VE 7.1 release has a lot of out-of-the-box features and can be used to run production workloads. If you want to play around with Proxmox, running the hypervisor inside a nested virtual machine in VMware ESXi is a great way to gain experience with installing, operating, troubleshooting, and other aspects of the virtualization solution.

You can learn more about Proxmox from their official page found here:

- Proxmox – Powerful open-source server solutions

Proxmox 8.1 New Features and Download with Software-Defined Network and Secure Boot

November 27, 2023
Proxmox

Proxmox 8.1

The Proxmox 8.1 hypervisor has been released with great new features. The official information and documentation show it is a worthy upgrade for Proxmox 8 systems. Highlights include new software-defined network (SDN) features, secure boot, flexible notifications, and other new improvements. Let's dive into this release.

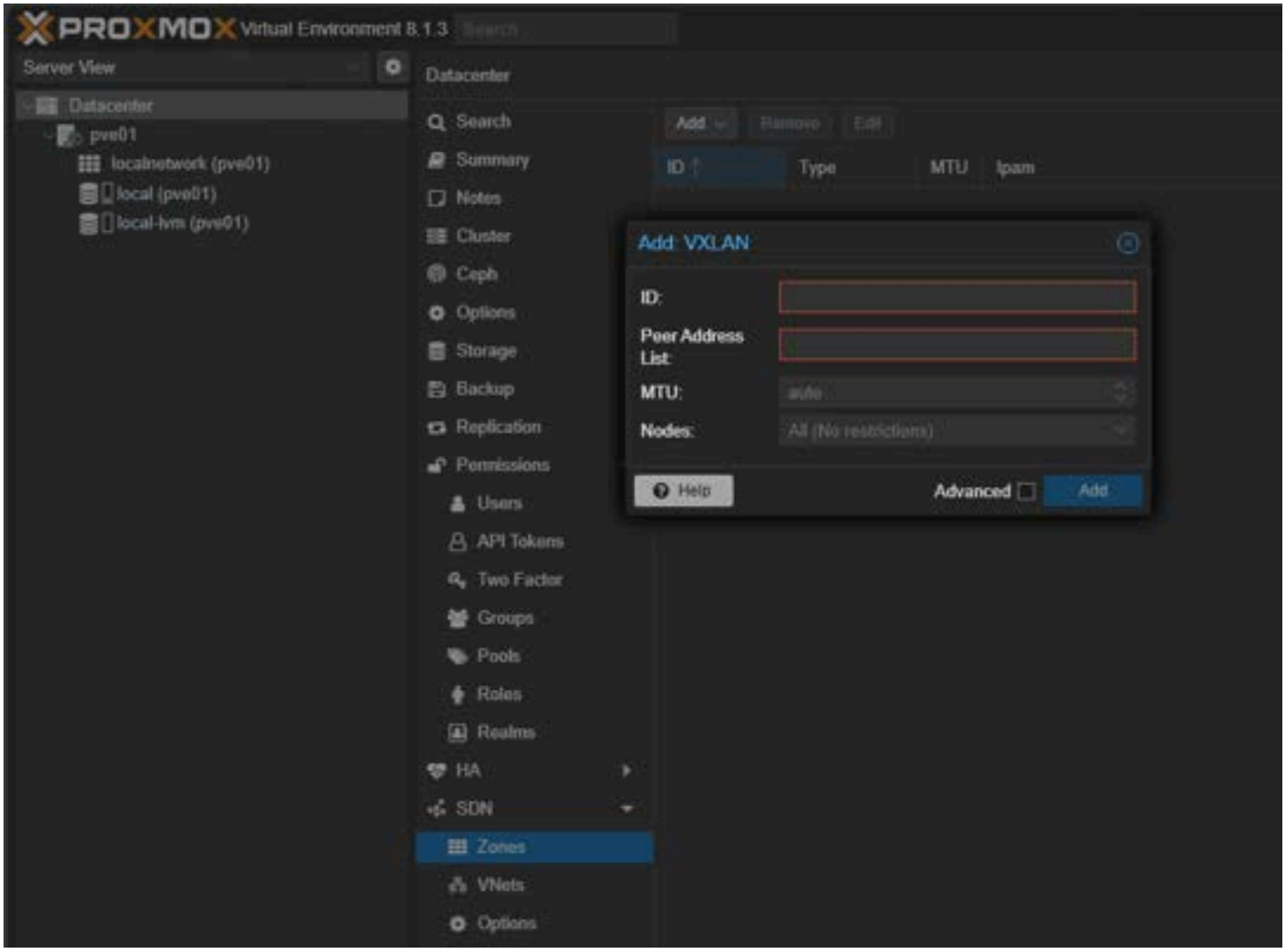
Table of contents

- Download and Community Support
- Proxmox is Open Source with Professional Support available
- Great for home labs
- Wrapping up new Proxmox VE 8.1 features

Software-Defined Networking in Proxmox VE 8.1

One of the top new features of Proxmox VE 8.1 is its native support for software-defined networking (SDN). Changes in this release, by default, the core SDN packages are now integrated into the Proxmox setup. This adds a more flexible, scalable networking solution within virtual environments and installations.

SDN in Proxmox VE 8.1 enables you to create virtual zones and networks, enabling you to manage and control complex networking configurations efficiently, right from the web interface. With this new feature, you can handle complex overlay networks and enhance multi-tenancy setups.

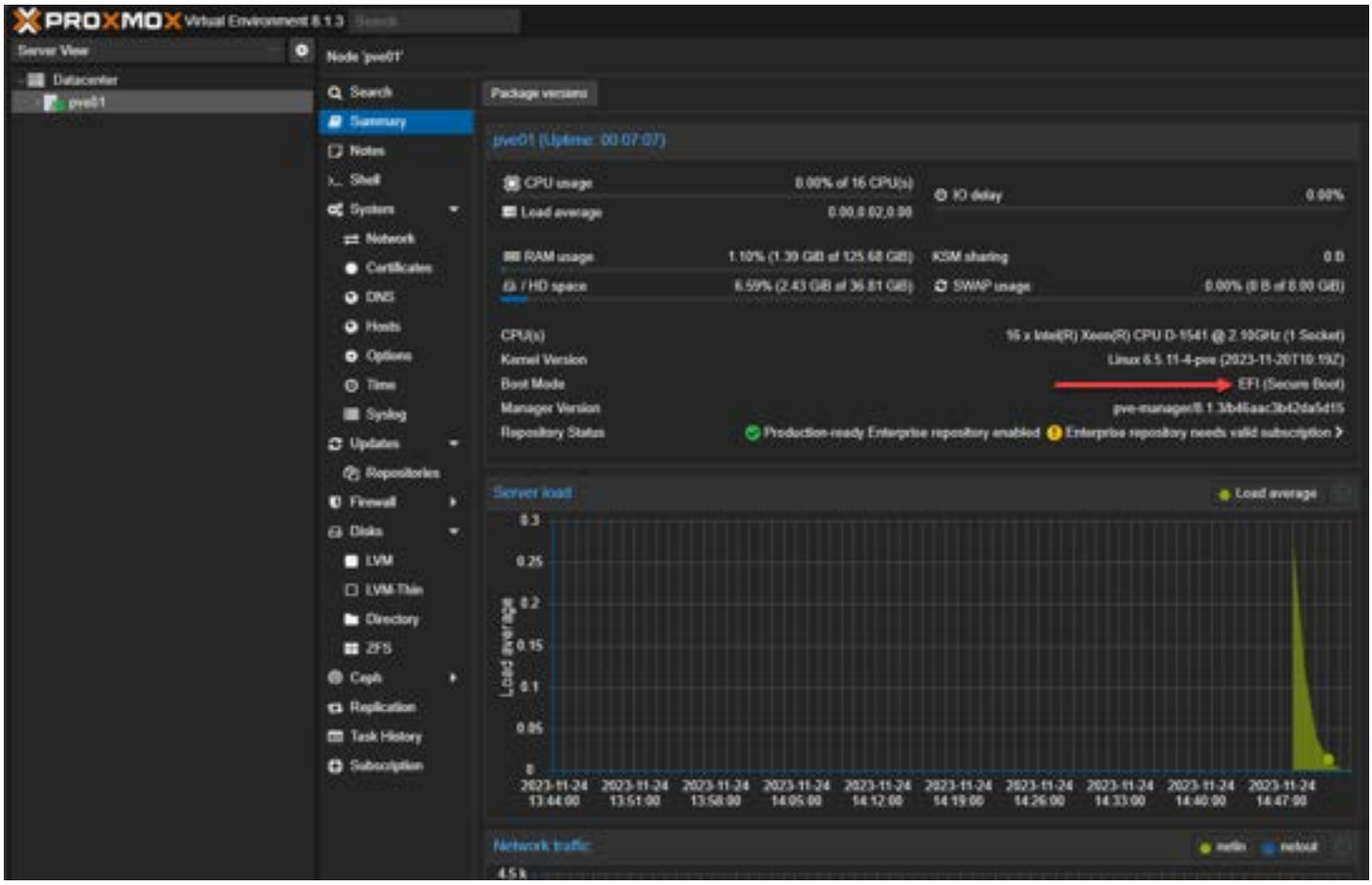


Software defined networking in proxmox 8.1

Enhancing Security with Secure Boot Compatibility

Security is enhanced in Proxmox VE 8.1 with the addition of support for Secure Boot. Secure Boot makes sure that only software with a valid digital signature is allowed to boot. This more secure boot process helps reduce the risk of unauthorized or malicious code execution in virtual machines.

Proxmox VE 8.1 now includes a signed shim bootloader, making it compliant with most hardware UEFI implementations. This feature is a great step forward in safeguarding virtualized data centers.

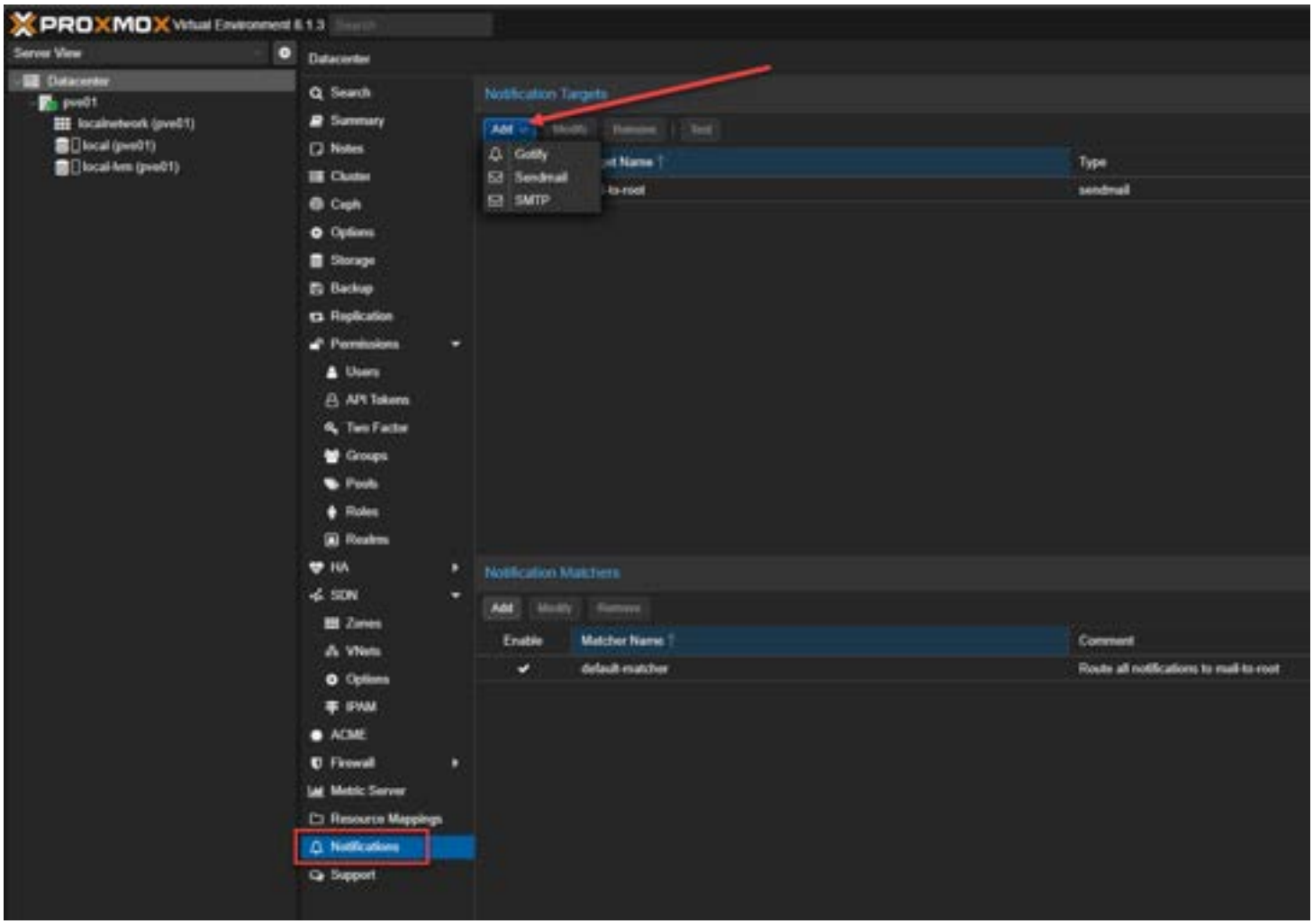


Efi secure boot enabled in proxmox 8.1

Introducing a Flexible Notification System support

Another new enhancement that many will be excited about is Proxmox VE 8.1 introduces a new, flexible notification system that employs a rules matcher-based approach to route notifications. This system allows users to specify various target types for receiving notifications.

It supports diverse notification channels, including local Postfix MTA, Gotify servers, and authenticated SMTP servers. The new granular control over notifications enhances system monitoring and response capabilities to system events.

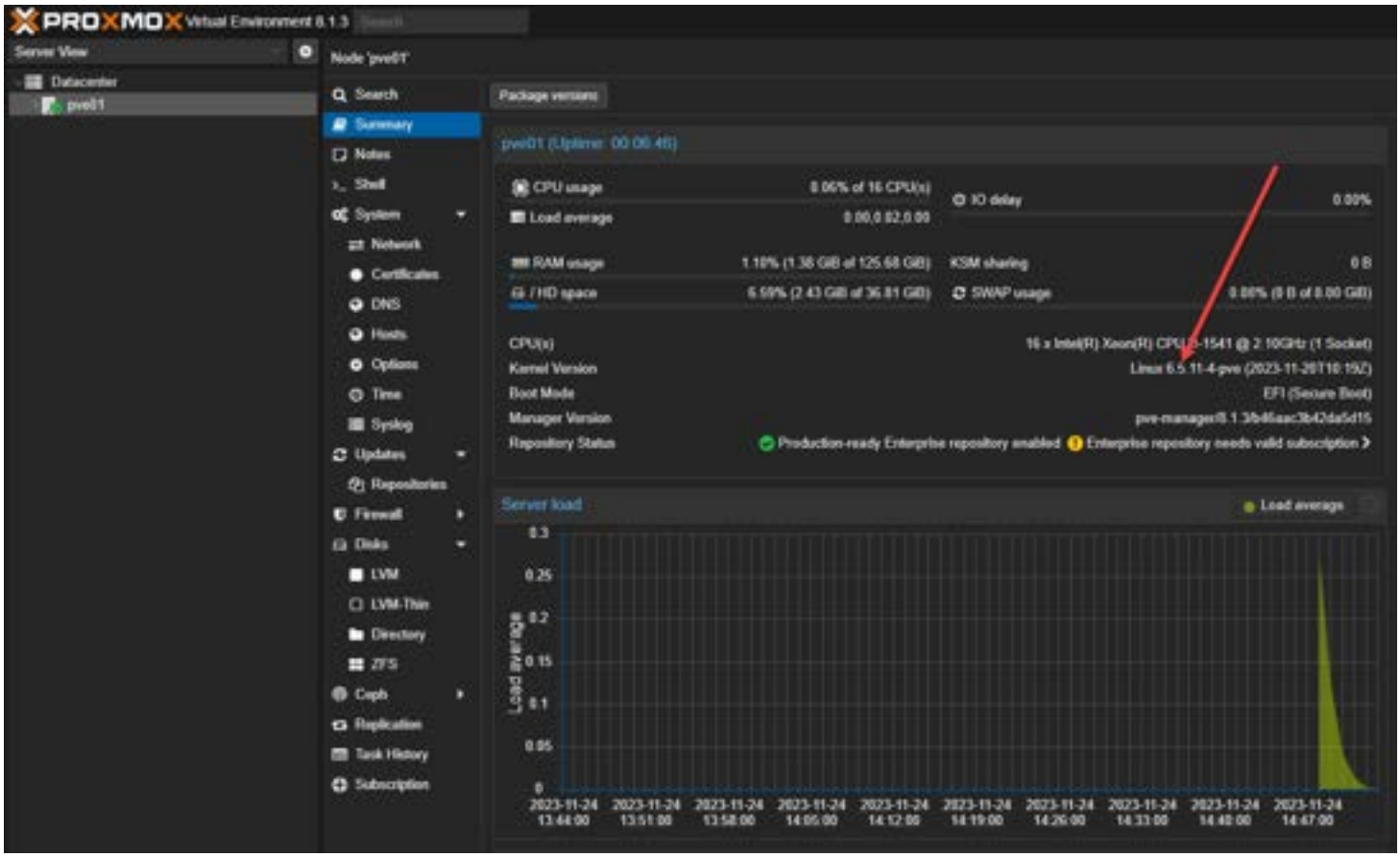


New notification system support

Kernel and Software Updates: Staying Ahead with Proxmox VE 8.1

The new release is based on Debian 12.2, codenamed “Bookworm,” and includes a newer Linux kernel 6.5. Keeping up with the latest kernel helps ensure stability and performance. Proxmox VE 8.1 also includes updates to open-source technologies, such as QEMU 8.1, Ceph 18.2, and Open ZFS 2.2.

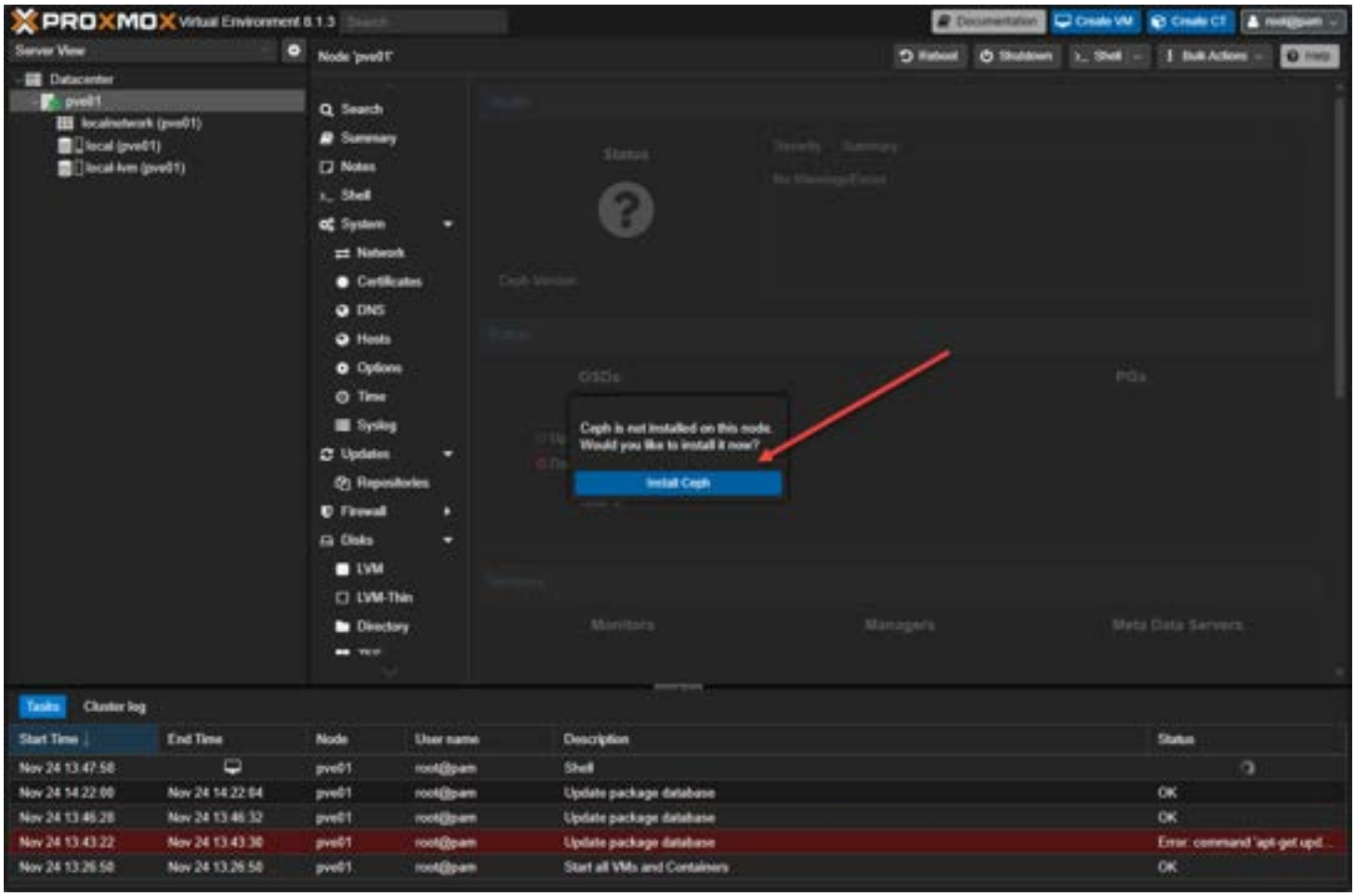
This will help to further enhance virtualization performance and storage technologies for virtualization tasks.



New linux kernel update with proxmox 8.1

Comprehensive Support for Ceph Versions

Proxmox VE 8.1 adds support for Ceph Reef 18.2.0 defaults and continues to provide compatibility with Ceph Quincy 17.2.7. This dual-version support provides flexibility in choosing the most appropriate Ceph version based on specific requirements and scenarios.



Installilng ceph in proxmox 8.1

Simplifying Virtual Machine Management

Proxmox VE 8.1 includes new bulk management features that make managing virtual machines more intuitive and efficient. It improves upon the “Bulk Actions” feature. These now include an option to suspend multiple guests simultaneously, adding new capabilities in streamlining administrative tasks.

Also, it adds a VirtIO driver ISO image that is now more straightforward and directly integrated into the VM creation wizard taking the heavy lifting out of this process.

Download and Community Support

Proxmox VE 8.1 is available for download from the official Proxmox website, complete with all features and capable of installation on bare-metal. The Proxmox community, with over 130,000 active members, continues to be a vibrant and supportive space for sharing knowledge and experiences.

Proxmox is Open Source with Professional Support available

As an open-source platform, Proxmox VE is licensed under the GNU Affero General Public License, v3, offering flexibility and freedom from vendor lock-in.

For enterprise users, Proxmox Server Solutions GmbH offers subscription-based support, ensuring access to tested updates and professional assistance.

Great for home labs

Many are already running Proxmox in their home lab environment. Proxmox is an excellent choice for home labbers who want a robust feature set for their lab VMs and self-hosted services and it is an open source virtualization platform. It

makes use of kernel based virtual machine (KVM).

The new Proxmox 8.1 features make it an even more appealing choice for running your critical self-hosted services. I have been running Proxmox in the home lab for a few years now alongside other hypervisors like vSphere. It is a great solution that allows you to run VMs and LXC containers without issue.

Proxmox 8 Cluster with Ceph Storage configuration
https://youtube.com/watch?v=-qk_P9SKYK4

Proxmox 8 cluster with Ceph storage

The web UI is fully-featured, and you can easily get to everything you need in the navigation links in the browser.

For me, I have had no major issues to report with great CPU performance and support for most project solutions I have installed. You can also passthrough your GPUs such as AMD and nVidia graphics cards. I you want to run a Docker container host, Proxmox makes for a great underlying hypervisor solution that you can also cluster with multiple hosts for HA, migration, and scalability purposes.

The Proxmox Backup server is also free to run and backup all your critical VM workloads. VM templates are available for quickly deploy various operating systems from the web-based console.

Wrapping up new Proxmox VE 8.1 features

Proxmox Virtual Environment 8.1 makes the Proxmox 8.x release even better with great new features and capabilities. The team at Proxmox is listening to what users and organizations need with Proxmox 8.1. Features like secure boot, SDWAN built-in, new kernel updates, better notification system and improved bulk operations, make this the best Proxmox VE release to date.

Upgrade Proxmox Host to 8.1: Tutorial & Steps

November 28, 2023
Proxmox

Proxmox 8.1 upgrade steps

With the release of Proxmox 8.1, you may be itching to update your Proxmox host in the home lab or production. Let's look at the steps to upgrade your Proxmox host to Proxmox 8.1. In the example below, I will be upgrading an 8.0.3 host that I have running to 8.1.

Table of contents

New features

There are many new features to speak of in Proxmox 8.1. I just uploaded a post covering the new features. However, as a quick overview, the major new features include:

- Software-defined networking
- Secure boot
- New bulk actions
- Upgraded Linux kernel
- A new flexible notification system
- Upgraded Ceph Reef version

No enterprise subscription prerequisites

If you are running Proxmox in the home lab and aren't running an enterprise subscription, which is how most home lab enthusiasts will be running, you need to reconfigure your update repositories. You may have already done this earlier for a lower-level Proxmox version. However, if you haven't already updated it to the "bookworm" repo, we will need to make that change, and then also change to the Ceph "reef" repo.

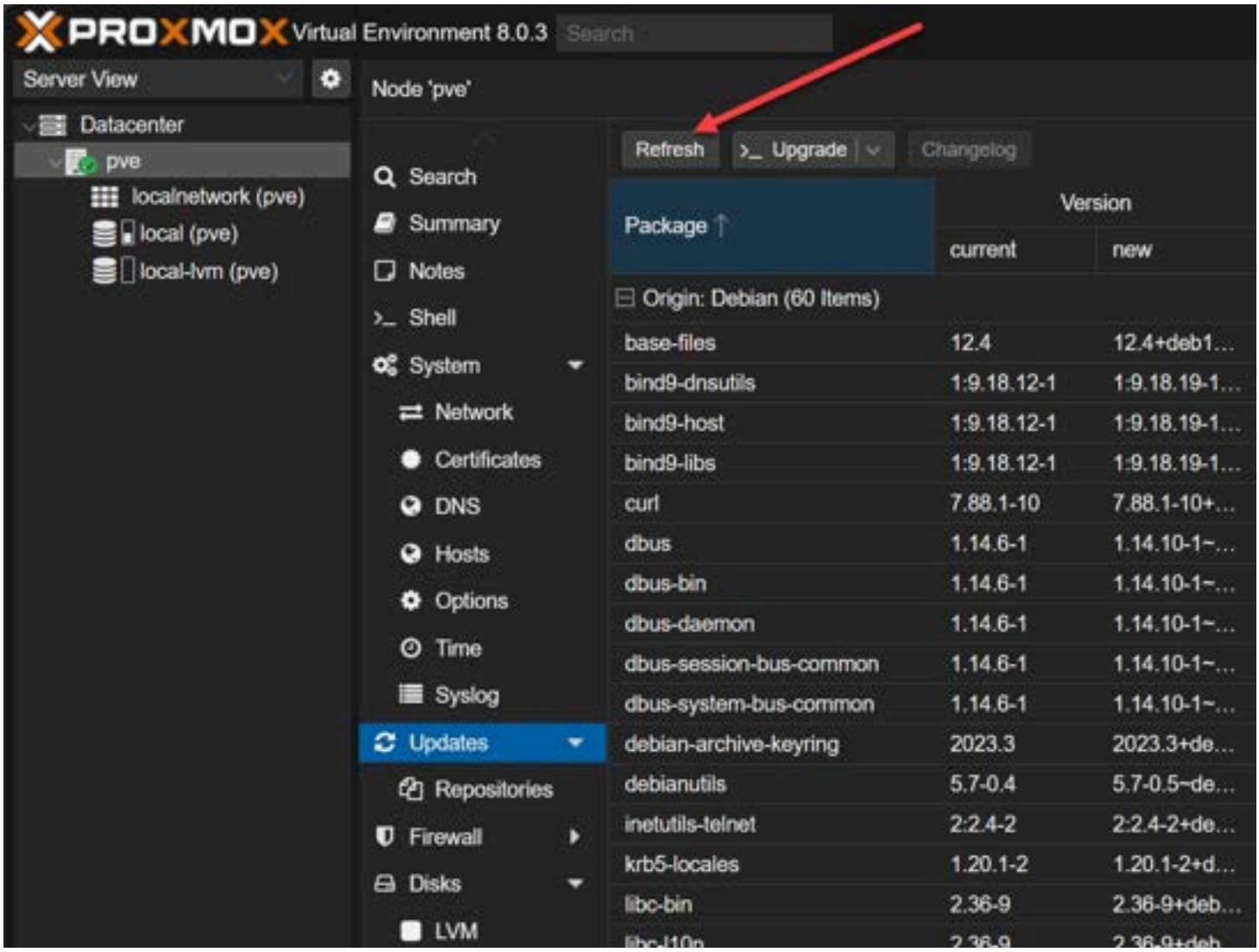
Update the following files in the comment lines:

```
#/etc/apt/sources.list.d/pve-enterprise.list

From: deb https://enterprise.proxmox.com/debian/pve bookworm enterprise
To: deb http://download.proxmox.com/debian/pve bookworm pve-no-subscription

#/etc/apt/sources.list.d/ceph.list
From: deb https://enterprise.proxmox.com/debian/ceph-quincy bookworm enterprise
To: deb http://download.proxmox.com/debian/ceph-reef bookworm no-subscription
```

Proxmox 8.1 upgrade steps from the GUI

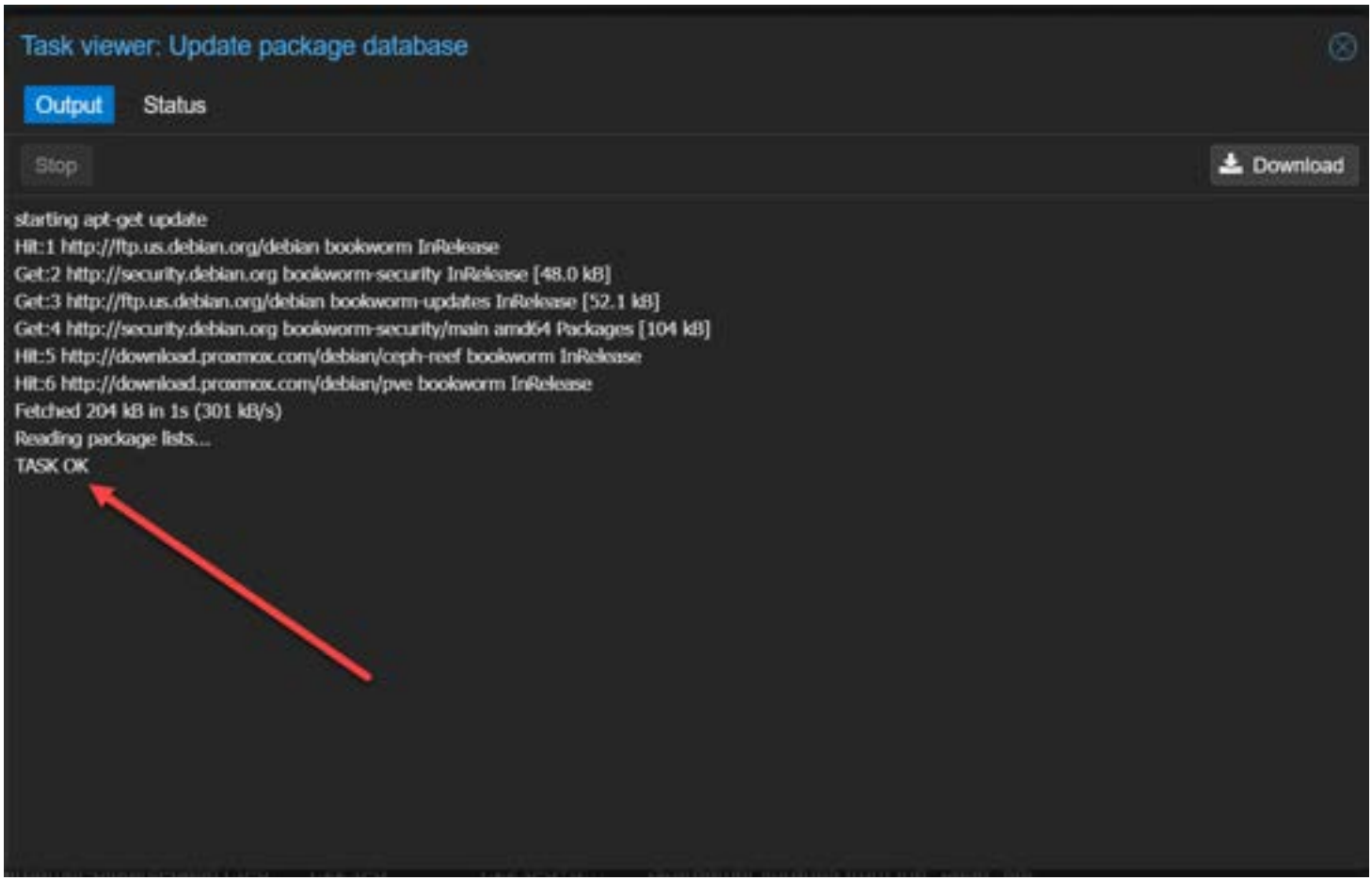
After you have reconfigured the files above, you will need to **refresh** your updates. The following are Proxmox 8.1 upgrade steps using the GUI web interface.

First, click your Proxmox host in the GUI. Navigate to **System > Updates > Refresh**. When you click **Refresh**, it runs an "apt-get update".



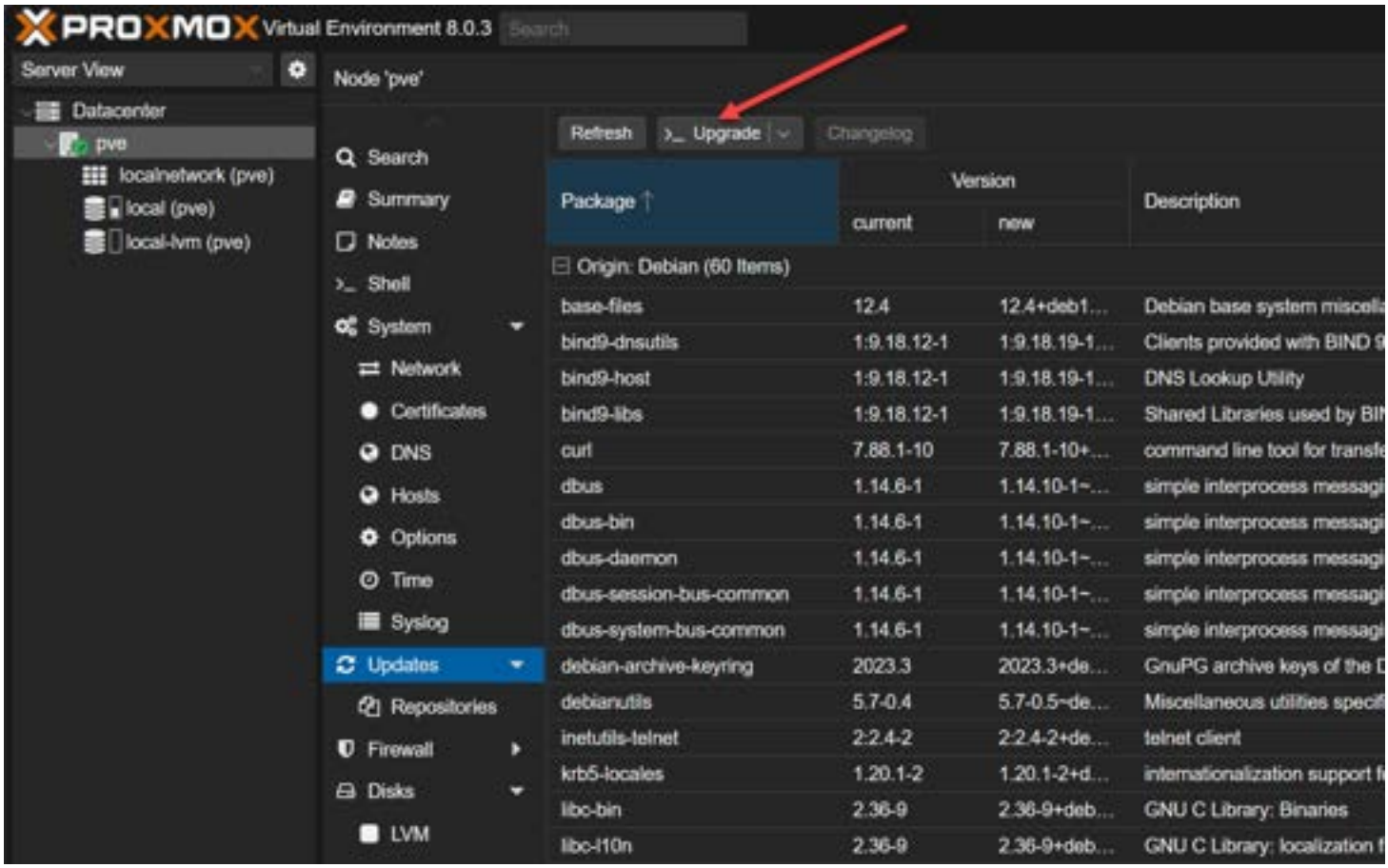
Refresh updates after changing the repositories

You will see the **Task viewer** display the status of the apt-get update.



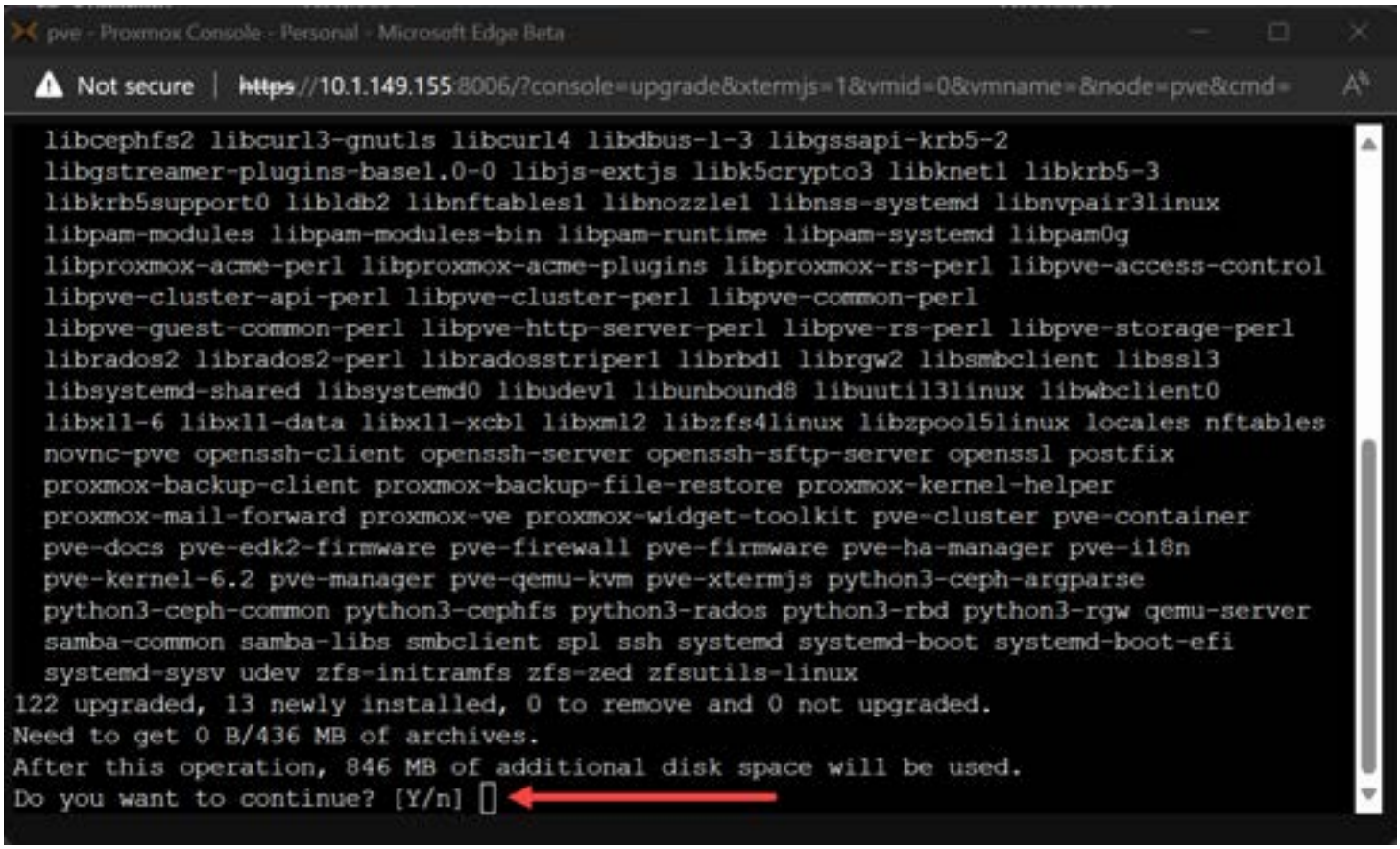
The status of the apt get update from the GUI

After refreshing the updates, you can click the **Upgrade** button.



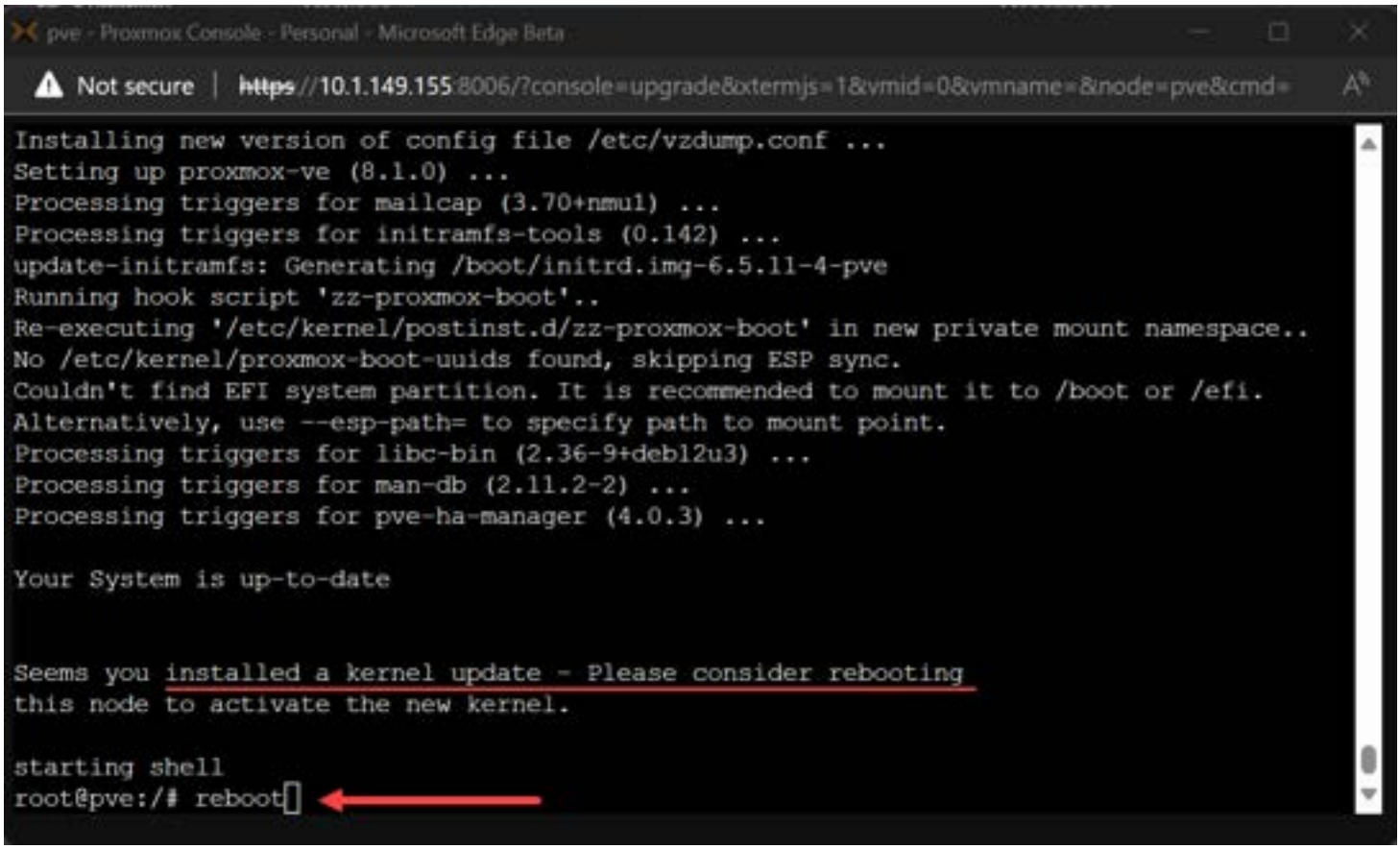
Kicking off the upgrade from the proxmox GUI

It will launch another browser window displaying the prompt for you to enter **Y** to confirm you want to continue the upgrade process.



Press y to continue the upgrade process

After all the upgrade process is complete, you will see the note that a new kernel was installed and a reboot is required to instantiate the new kernel. Here I am typing **reboot** from the window.



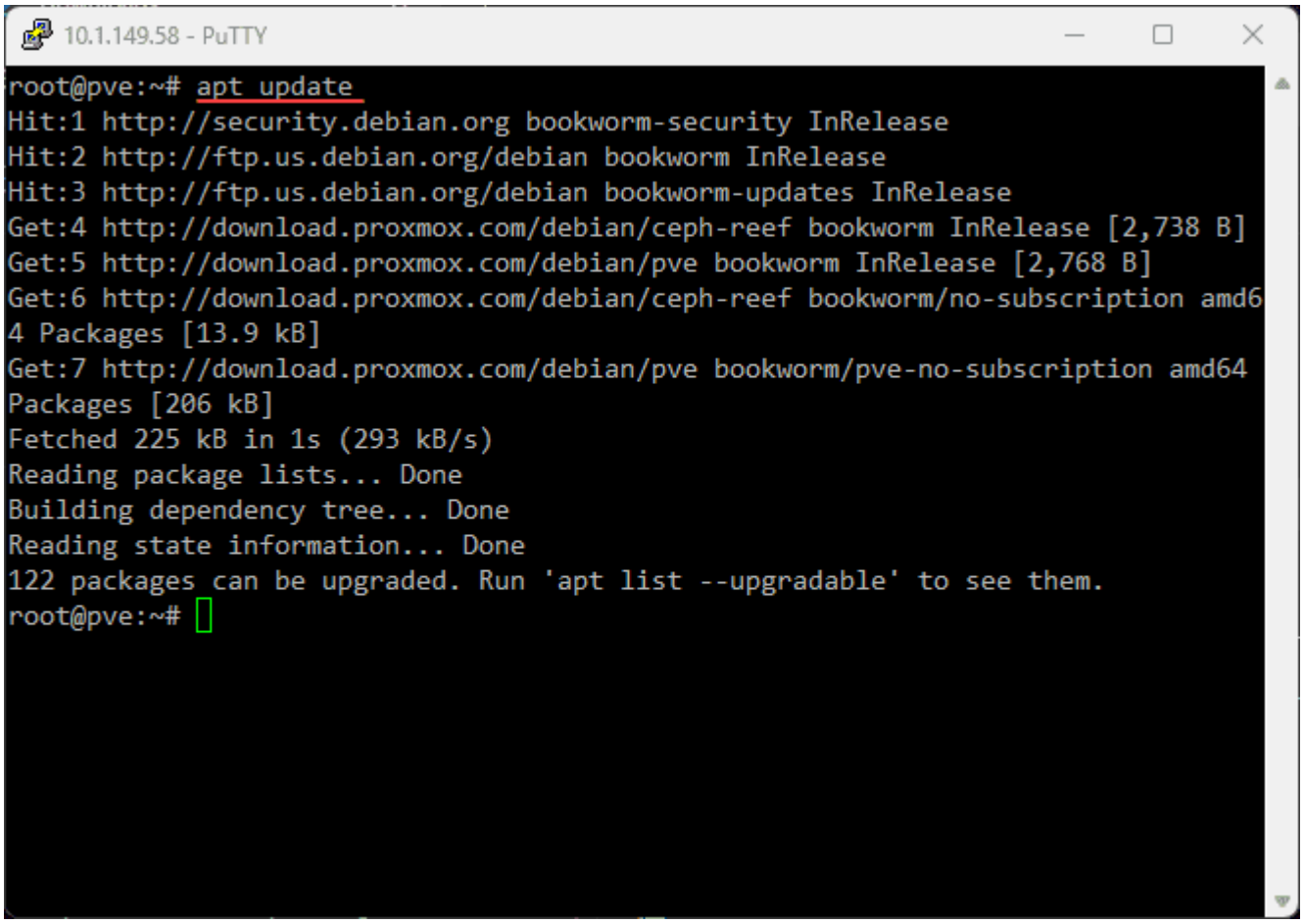
Reboot after the proxmox 8.1 upgrade and the kernel upgrade

Proxmox 8.1 upgrade steps from the command line

The upgrade steps from the command line are very simple. We just run the commands the GUI runs for us from the command line.

First we **refresh** the updates after we have updated the repository URLs. To do that, run the following commands:

```
apt update
```



Running the apt update command to refresh the available updates

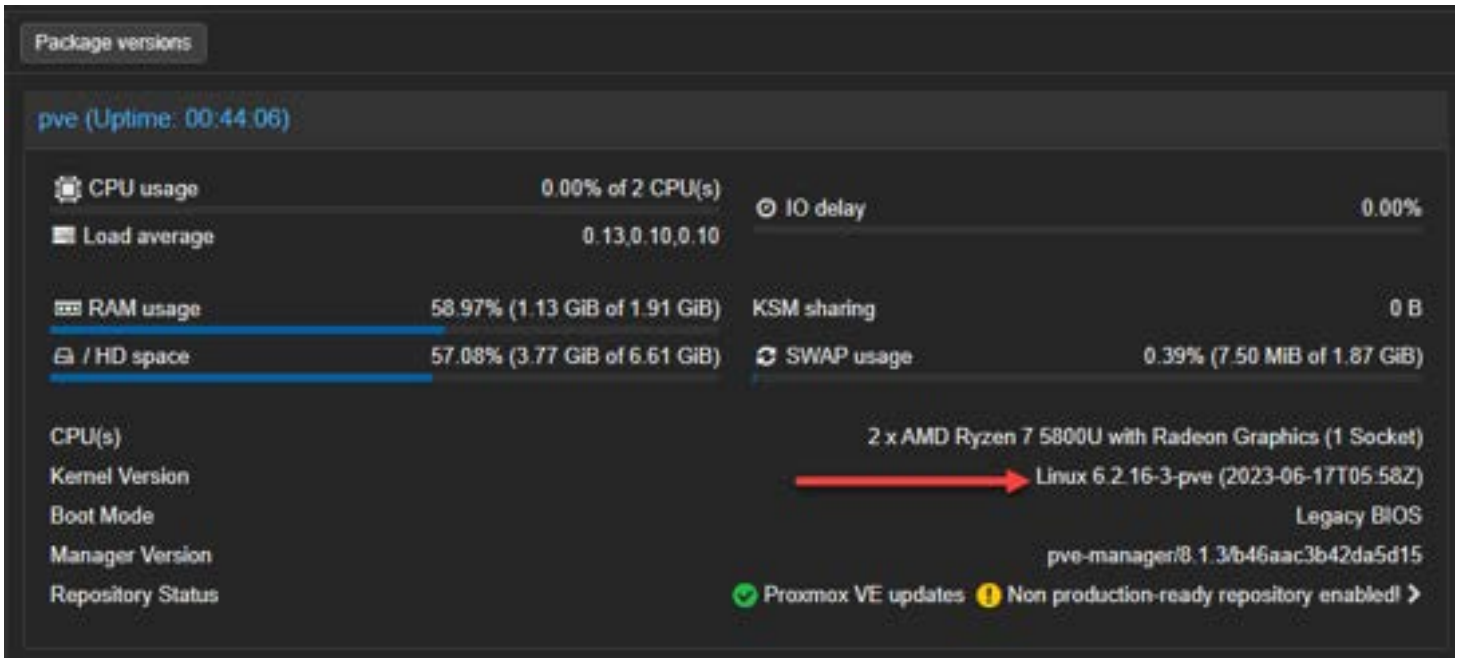
Next, we run the following command:

```
apt dist upgrade
```

```
root@pve:~# apt dist-upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following package was automatically installed and is no longer required:
  pve-kernel-6.2
Use 'apt autoremove' to remove it.
The following NEW packages will be installed:
  fonts-font-logos libnet-subnet-perl libpve-network-perl libpve-notify-perl
  libsocket6-perl proxmox-default-kernel proxmox-kernel-6.2
  proxmox-kernel-6.2.16-19-pve proxmox-kernel-6.5
  proxmox-kernel-6.5.11-4-pve-signed proxmox-termproxy
  pve-edk2-firmware-legacy pve-edk2-firmware-ovmf
The following packages will be upgraded:
  base-files bind9-dnsutils bind9-host bind9-libs ceph-common ceph-fuse curl
  dbus dbus-bin dbus-daemon dbus-session-bus-common dbus-system-bus-common
  debian-archive-keyring debianutils grub-common grub-efi-amd64-bin grub-pc
  grub-pc-bin grub2-common ifupdown2 inetutils-telnet krb5-locales libc-bin
  libc-l10n libc6 libcephfs2 libcurl3-gnutls libcurl4 libdbus-1-3
  libgssapi-krb5-2 libgstreamer-plugins-base1.0-0 libjs-extjs libk5crypto3
  libknet1 libkrb5-3 libkrb5support0 libldb2 libnftables1 libnozzle1
  libnss-systemd libnvpair3linux libpam-modules libpam-modules-bin
  libpam-runtime libpam-systemd libpam0g libproxmox-acme-perl
  libproxmox-acme-plugins libproxmox-rs-perl libpve-access-control
  libpve-cluster-api-perl libpve-cluster-perl libpve-common-perl
  libpve-guest-common-perl libpve-http-server-perl libpve-rs-perl
  libpve-storage-perl librados2 librados2-perl libradosstriper1 librbd1
  librgw2 libsmbclient libssl3 libsystemd-shared libsystemd0 libudev1
  libunbound8 libuutil3linux libwbclient0 libx11-6 libx11-data libx11-xcb1
  libxml2 libzfs4linux libzpool5linux locales nftables novnc-pve
  openssh-client openssh-server openssh-sftp-server openssl postfix
  proxmox-backup-client proxmox-backup-file-restore proxmox-kernel-helper
  proxmox-mail-forward proxmox-ve proxmox-widget-toolkit pve-cluster
  pve-container pve-docs pve-edk2-firmware pve-firewall pve-firmware
  pve-ha-manager pve-i18n pve-kernel-6.2 pve-manager pve-qemu-kvm pve-xtermjs
  python3-ceph-argparse python3-ceph-common python3-cephfs python3-rados
  python3-rbd python3-rgw qemu-server samba-common samba-libs smbclient spl
  ssh systemd systemd-boot systemd-boot-efi systemd-sysv udev zfs-initramfs
  zfs-zed zfsutils-linux
122 upgraded, 13 newly installed, 0 to remove and 0 not upgraded.
Need to get 436 MB of archives.
After this operation, 846 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Running the apt dist upgrade

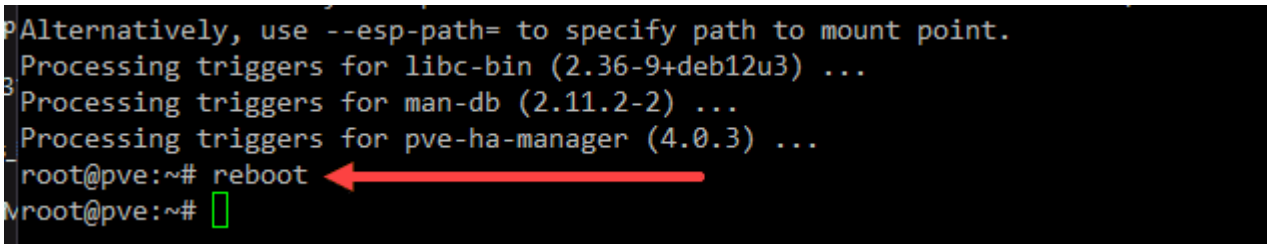
After the upgrade is successful from the command line, if you look at your Proxmox host summary, you will see it has upgraded to 8.1.3, but the Linux kernel is still at version **6.2**. So, we need to reboot.



Before we reboot the kernel still shows 6.2

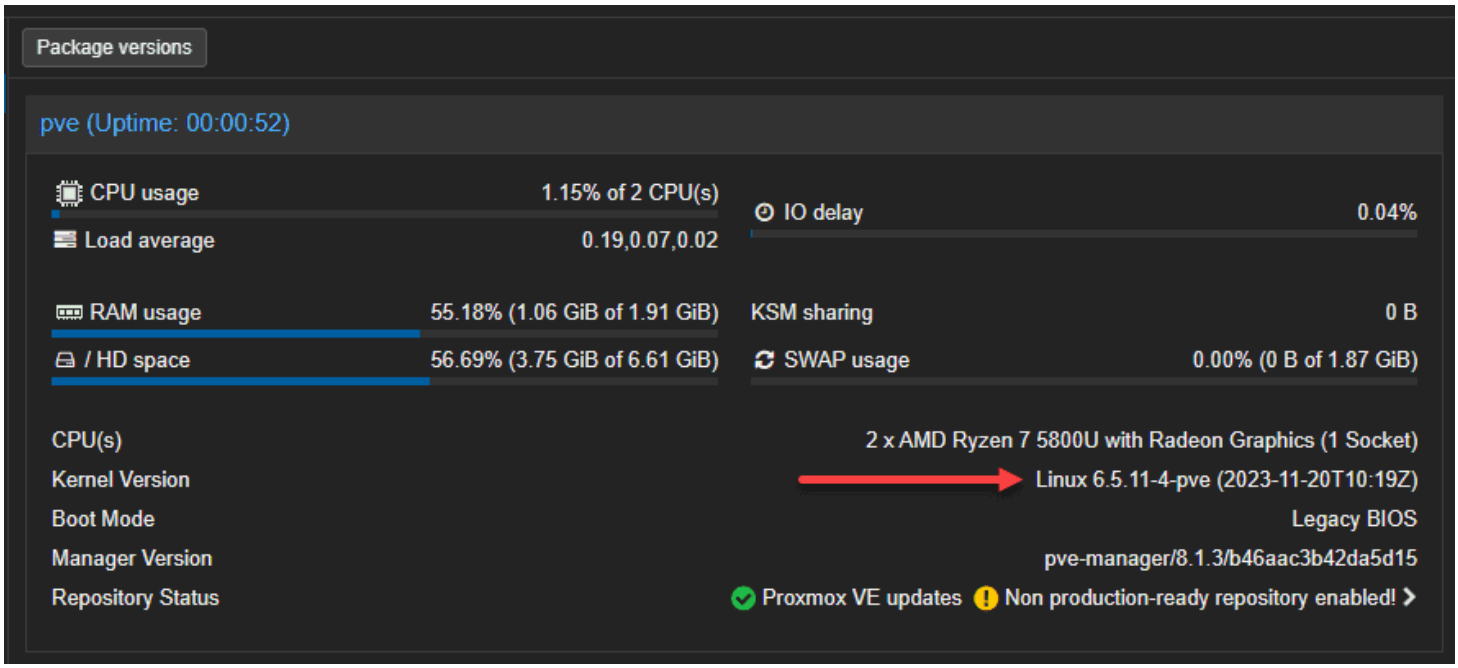
From the command line issue the reboot command:

```
reboot
```



Running the reboot command to reboot proxmox and install the new kernel

Now, we can check the kernel version again and we see the Linux 6.5 kernel has been installed.



After the reboot the new linux 6.5 kernel has been installed

Steps to upgrade from Proxmox 7.4 to Proxmox 8.1

The steps to upgrade from Proxmox 7.4 to Proxmox 8.1 are fairly straightforward. However, it does involve more steps if you are currently running Ceph Quincy.

First, you will need to upgrade Ceph from Pacific to Quincy. The next step involves upgrading Proxmox VE from version 7.4 to 8.1. In the last step, once you have Proxmox VE 8.1 running, you will upgrade your Ceph installation to Reef.

Here are the links to the official documentation on those specific steps:

- Ceph Pacific to Quincy Upgrade Guide
- Upgrading from Proxmox VE 7 to 8
- Ceph Quincy to Reef Upgrade Guide

Frequently Asked Questions About Upgrading to Proxmox VE 8.1

How do I upgrade to Proxmox VE 8.1 from an older version?

Upgrading to Proxmox VE 8.1 can be achieved through the 'apt' command line tool. It's important to make sure that your current system is up to date before starting the upgrade. Detailed steps and guidance are available in the Proxmox VE documentation.

Is there support for migrating virtual machines in Proxmox VE 8.1?

Yes, Proxmox VE 8.1 supports migrating virtual machines. You can use Proxmox's built-in tools to move VMs between hosts, even across different versions, with minimal downtime.

How does the new SDN feature in Proxmox VE 8.1 impact network configuration?

The software-defined network (SDN) feature in Proxmox VE 8.1 allows for more adaptable network infrastructure configurations. You can now manage complex networking configurations more effectively, including creating virtual zones for improved network isolation.

Can I manage Proxmox VE 8.1 using the web-based user interface?

Proxmox VE 8.1 continues to offer a great web UI that provides easy management of virtual machines, containers, and network settings.

Are there any special considerations for Proxmox VE 8.1 with Ceph storage solutions?

Proxmox VE 8.1 supports Ceph Reef 18.2.0 and Ceph Quincy 17.2.7.

Does Proxmox VE 8.1 offer any enhancements in managing Linux containers?

Proxmox VE 8.1 has an updated kernel and software stack and provides improved support for Linux containers (LXC). The new kernel offers enhanced performance and stability for containerized applications.

How does the newer Linux kernel in Proxmox VE 8.1 benefit users?

The newer Linux Kernel 6.5 in Proxmox VE 8.1 brings many new improvements. These include performance benefits, better hardware support, and enhanced security features. This helps to provide a more efficient and secure virtual environment.

What are the best practices for backup and recovery in Proxmox VE 8.1?

You can easily use Proxmox's backup tools to schedule and manage backups effectively. Backups should be a regular part of your infrastructure, even in the home lab environment. Backups help make sure you can recover quickly if you have a hardware failure or accidental data deletion.

Mini PC running Proxmox

If you are looking for something to run Proxmox, you can easily install it on a Mini PC to get your feet wet in the home lab. Check out the video below:

Proxmox Mini Server: Beelink Mini PC S12 Pro
https://youtube.com/watch?v=xA_nnGO7HzA

Wrapping up the Proxmox 8.1 upgrade steps

As shown, the upgrade to Proxmox 8.1 can be accomplished using the Proxmox GUI and the command line. The steps involve changing the update repositories if you aren't running with a Proxmox subscription. Once you have the repositories updated, you can refresh the updates and install the available updates, including the upgrade to Proxmox 8.1.

Proxmox Networking for VMware vSphere admins

December 28, 2023
Proxmox

Proxmox networking for vsphere admins

One of the challenges that we run into when we are more familiar with one vendor over another is the difference in the technologies, how they work for customers, what they are called, and how to configure them. On the networking side of things, this can be the case as well. If you are familiar with VMware vSphere and looking to work with and play around with Proxmox, Proxmox networking may be foreign to work with when trying to compare it with VMware ESXi networking, etc. In this Proxmox networking for vSphere admins post for the community, we will look at the equivalent networking configurations in Proxmox compared to vSphere.

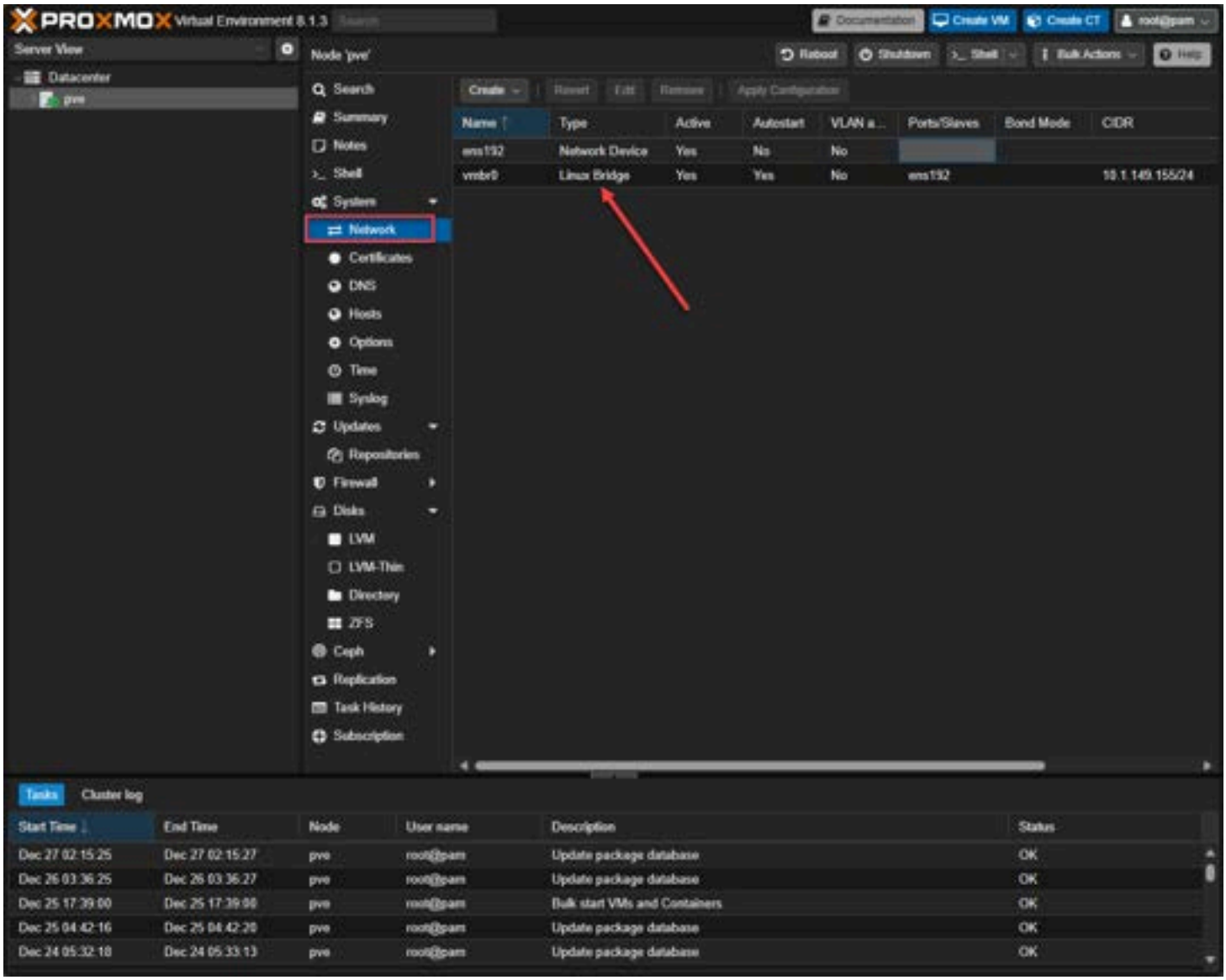
Table of contents

1. Proxmox Linux Bridge equivalent to the VMware vSwitch

The most equivalent network construct out of the box with Proxmox is the default Proxmox Linux bridge in the Proxmox environment. With the Linux Bridge in Proxmox, you establish the initial connectivity to your Proxmox host with a management IP address. Also, the default Linux Bridge is backed by a physical network adapter(s).

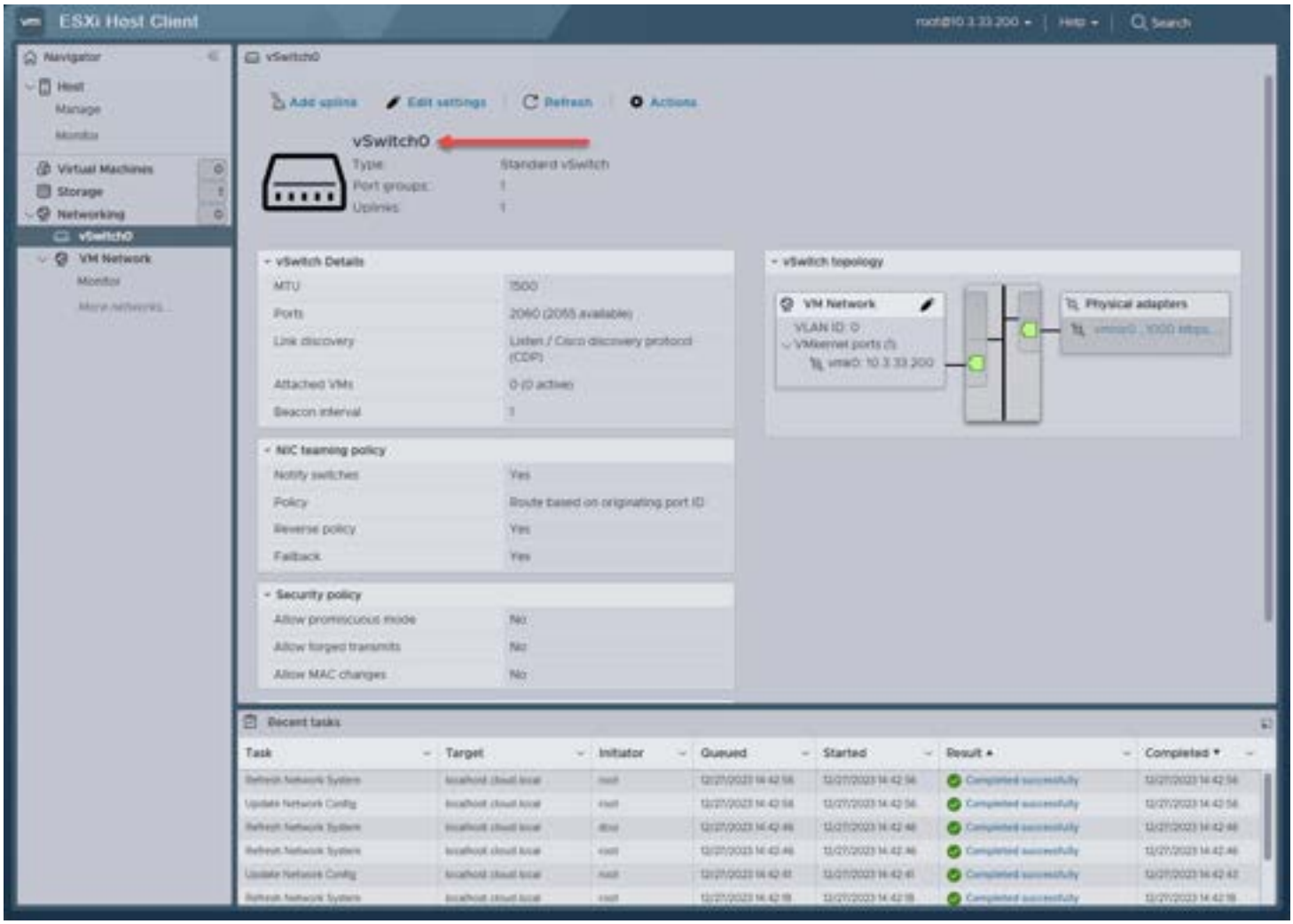
Below:

- Ports/Slaves – This shows the physical network adapter **ens192** assigned to the default Linux bridge
- CIDR – Shows the IP address and mask associated with the Linux bridge



Viewing the default proxmox linux bridge

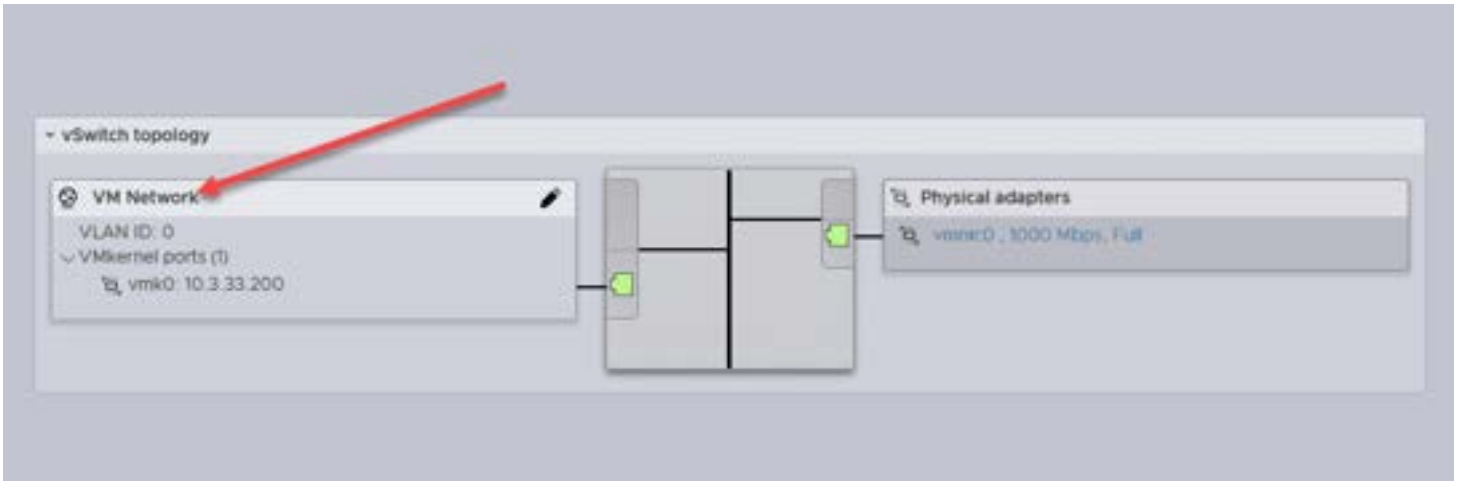
This is very similar to the default **vSwitch0** created in VMware ESXi right out of an install. As you can see below, we have a physical network adapter backing vSwitch0.



Vmware esxi default vswitch0

2. Proxmox Linux VLANs equivalent to VMware vSwitch Port Groups

As most vSphere admins are aware, you have the vSwitch0 and the default **VM Network port group** as shown above. Port groups in [vSphere allow you to create VLAN-tagged network labels in vSphere to assign VLAN tags to the virtual machines]() connected to them.

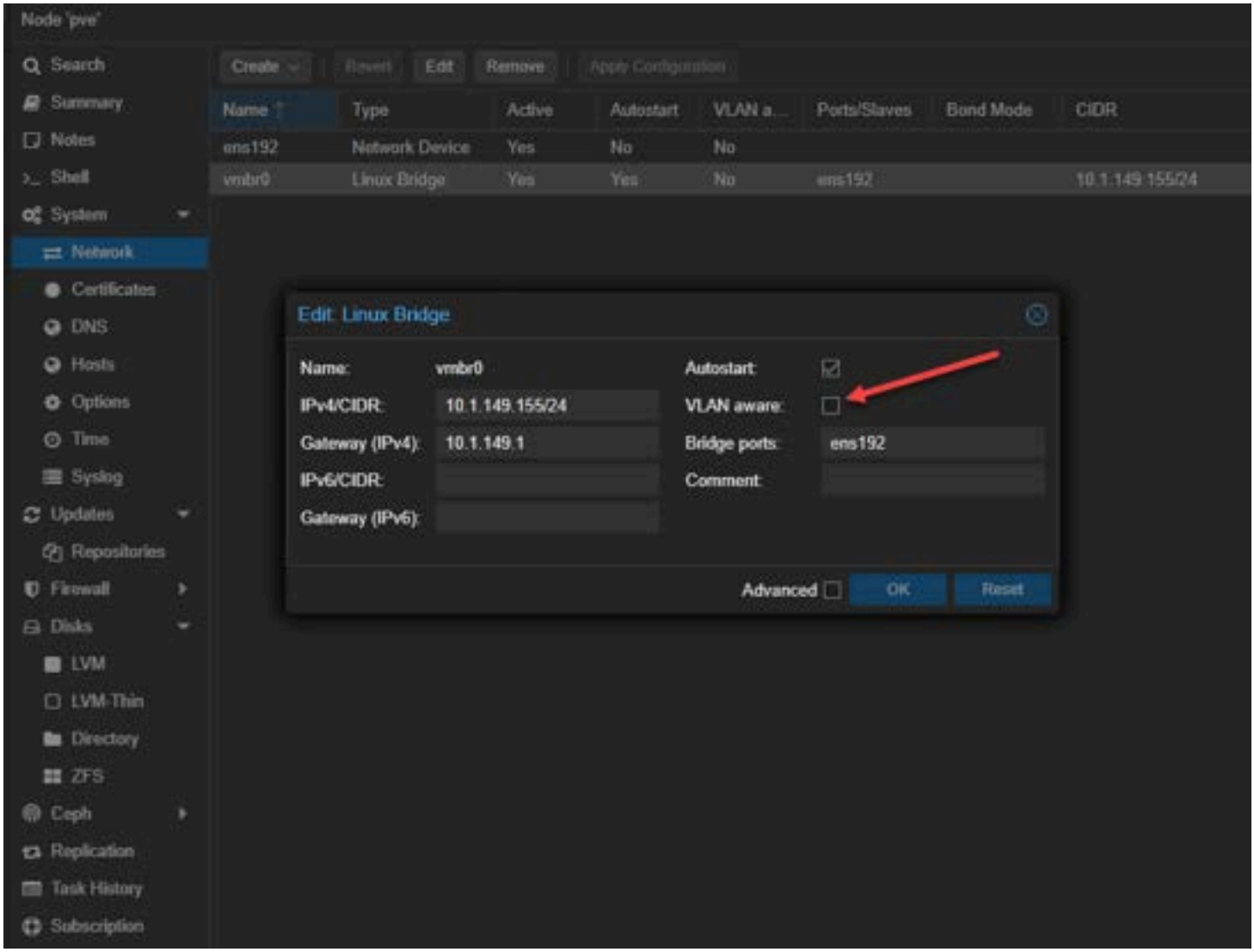


Default vsphere port group on vswitch0

In Proxmox networking with the default Linux Bridge, the port group construct carries over to what is known as a **Linux VLAN**. By default, when you create a VM in Proxmox without any other configuration, you can attach the VM to the default **vmbr0** bridge, which is essentially **VLAN 0** and assumes untagged traffic.

Unlike the VMware default vSwitch0 and VM Network port group, the default Proxmox Linux Bridge is not VLAN-aware out of the box. You have to enable this.

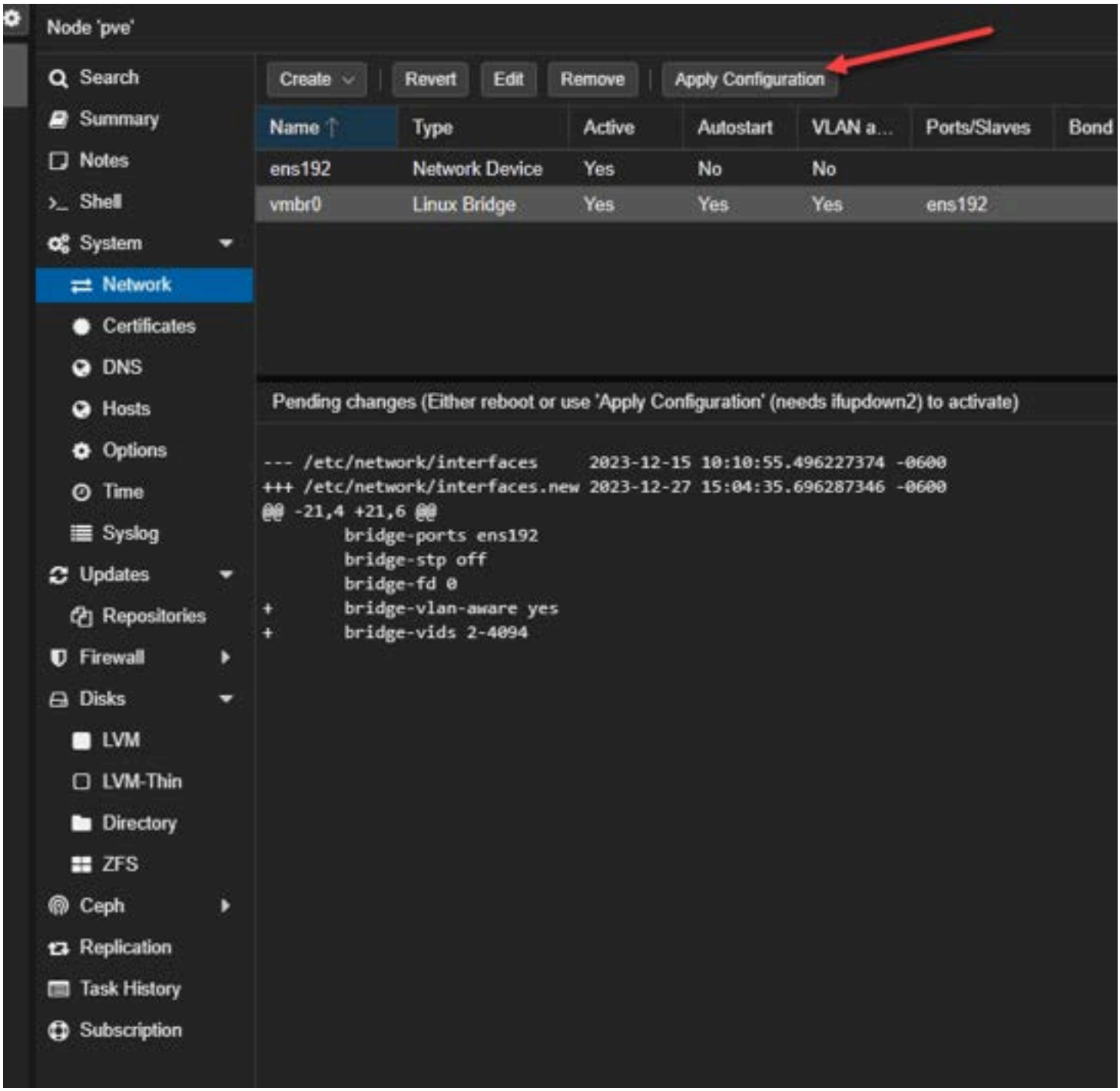
When you edit the default Linux Bridge, you will see the checkbox **VLAN aware** available on the Linux Bridge properties. Also, you will see basic networking configurations like the IP address and subnet, gateway for routing, etc. Place a check box in the VLAN aware checkbox.



Making the proxmox linux bridge vlan aware

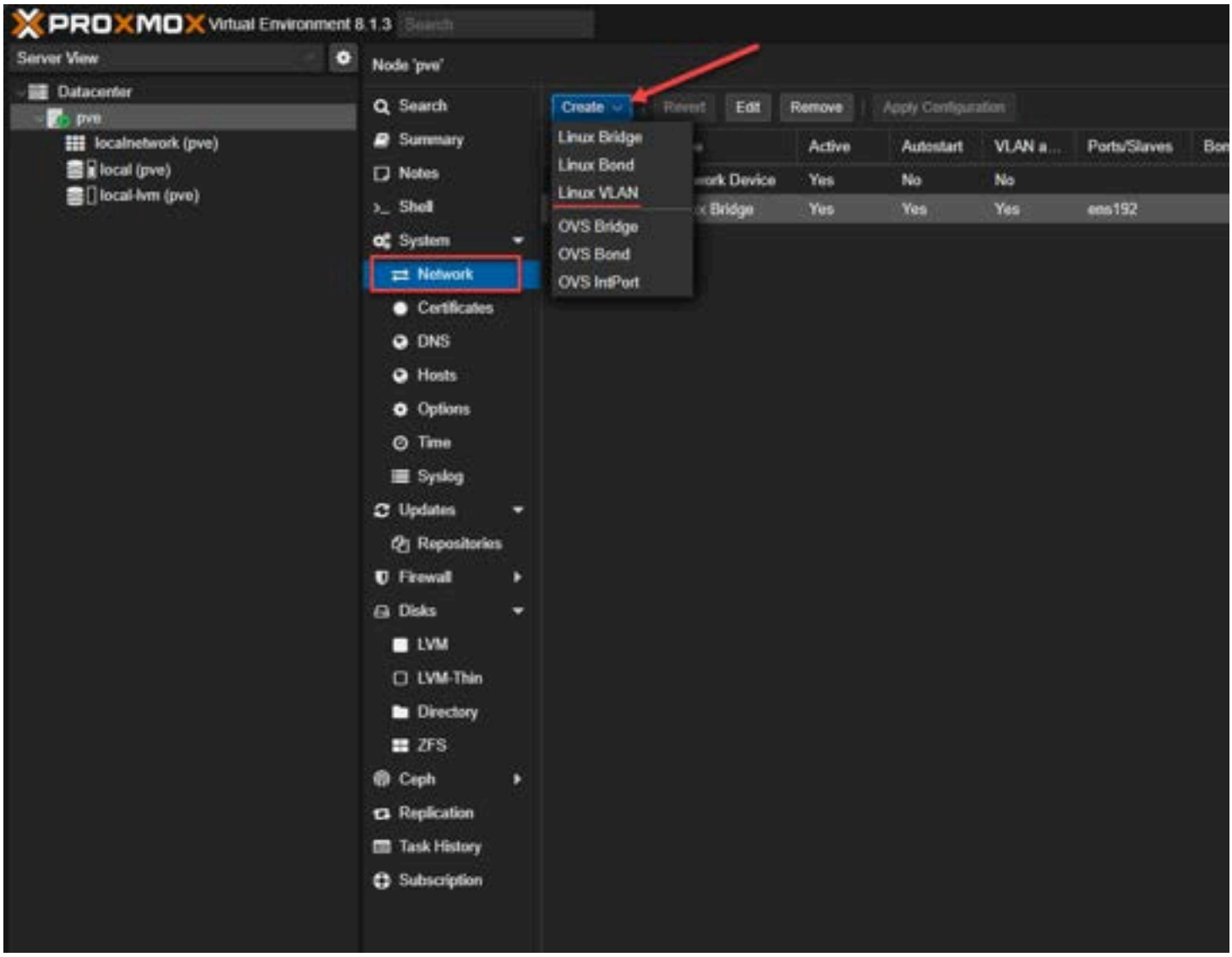
Now we can apply the configuration. Click the **Apply Configuration** button. Also, in the preview of the Pending changes, you will see the new VLAN bridge-ports configuration set to auto, containing the configuration lines:

```
bridge-ports ens192
bridge-stp off
bridge-vlan-aware yes
bridge-vids 2-4094
```



Apply the vlan aware configuration

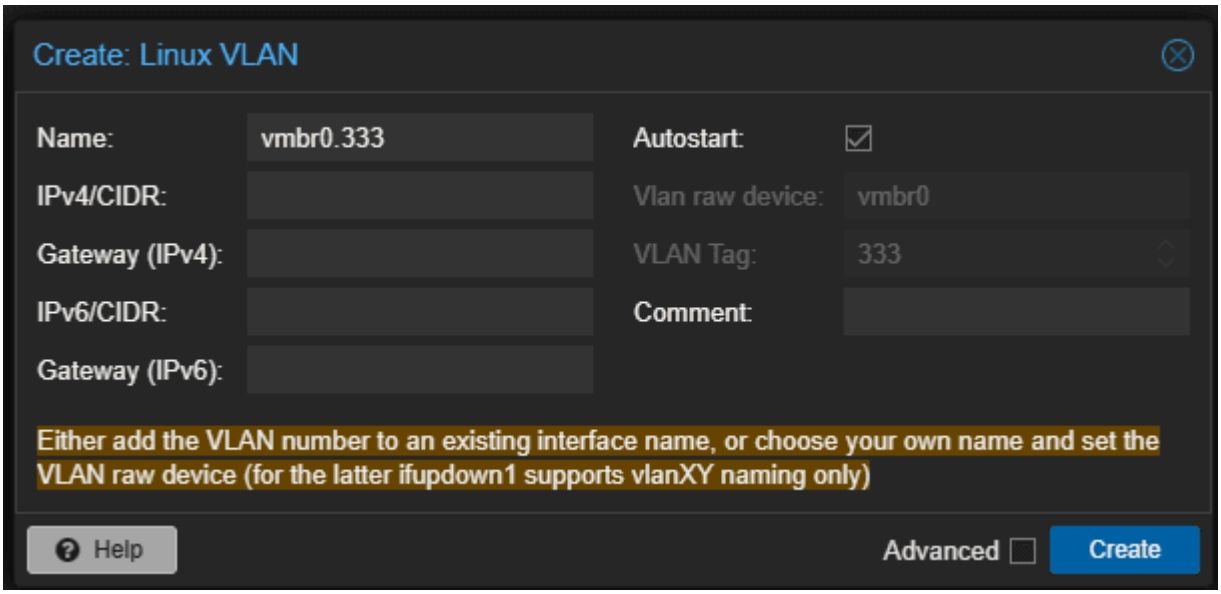
This configuration essentially makes the default Linux Bridge able to understand VLANs and VLAN traffic, so we can add Linux VLANs.



Creating a linux vlan

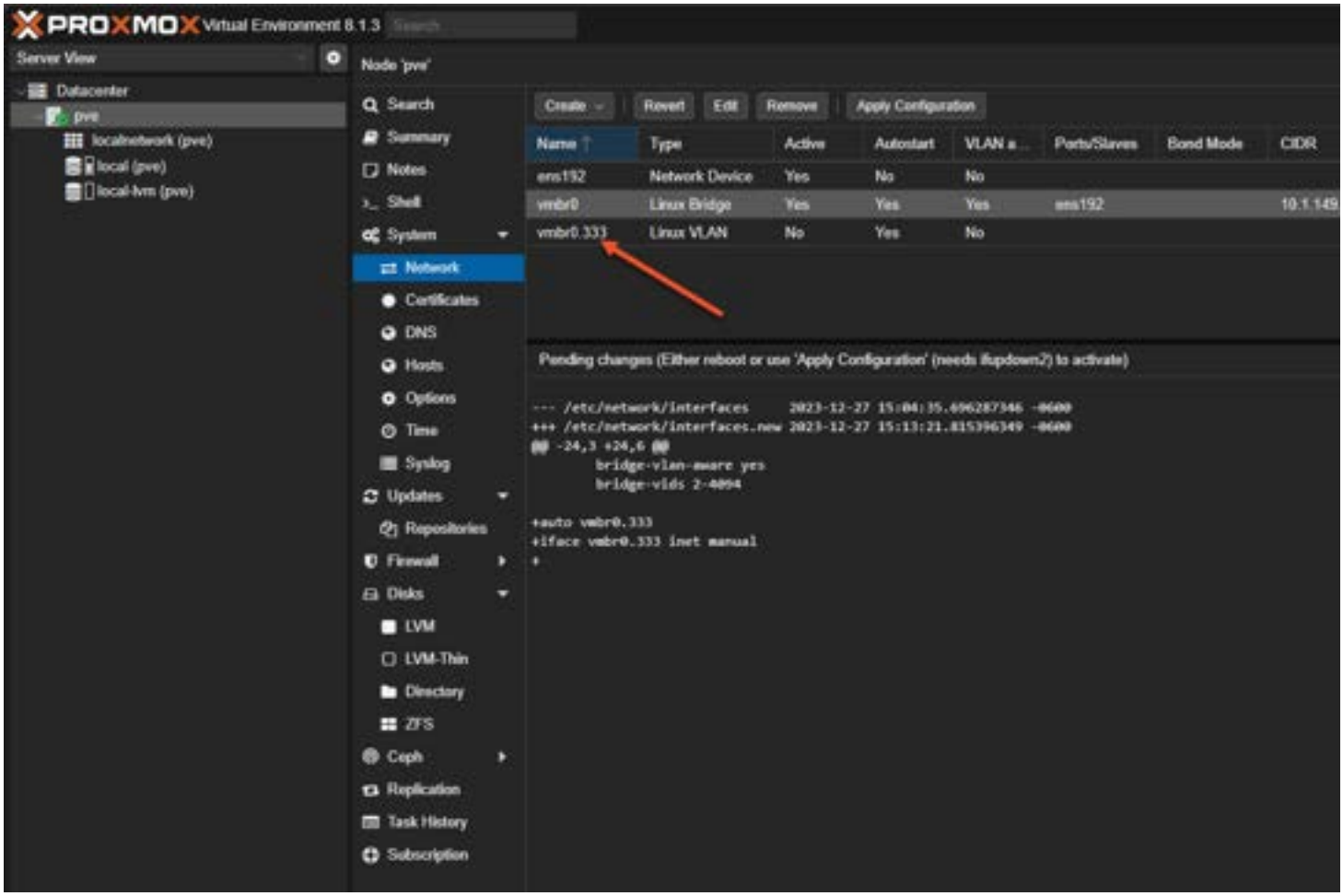
Now we can populate the new Linux VLAN with the appropriate configuration. Once you name the VLAN with the parent **vmbr0** interface, you will see the **VLAN raw device** and **VLAN Tag** greyed out. This essentially says we are creating a new Linux VLAN on the parent Linux Bridge interface, vmbr0.

Under the **Advanced** checkbox, you can set the MTU value in case you are wondering.



Creating the linux vlan from the vmbr0 interface

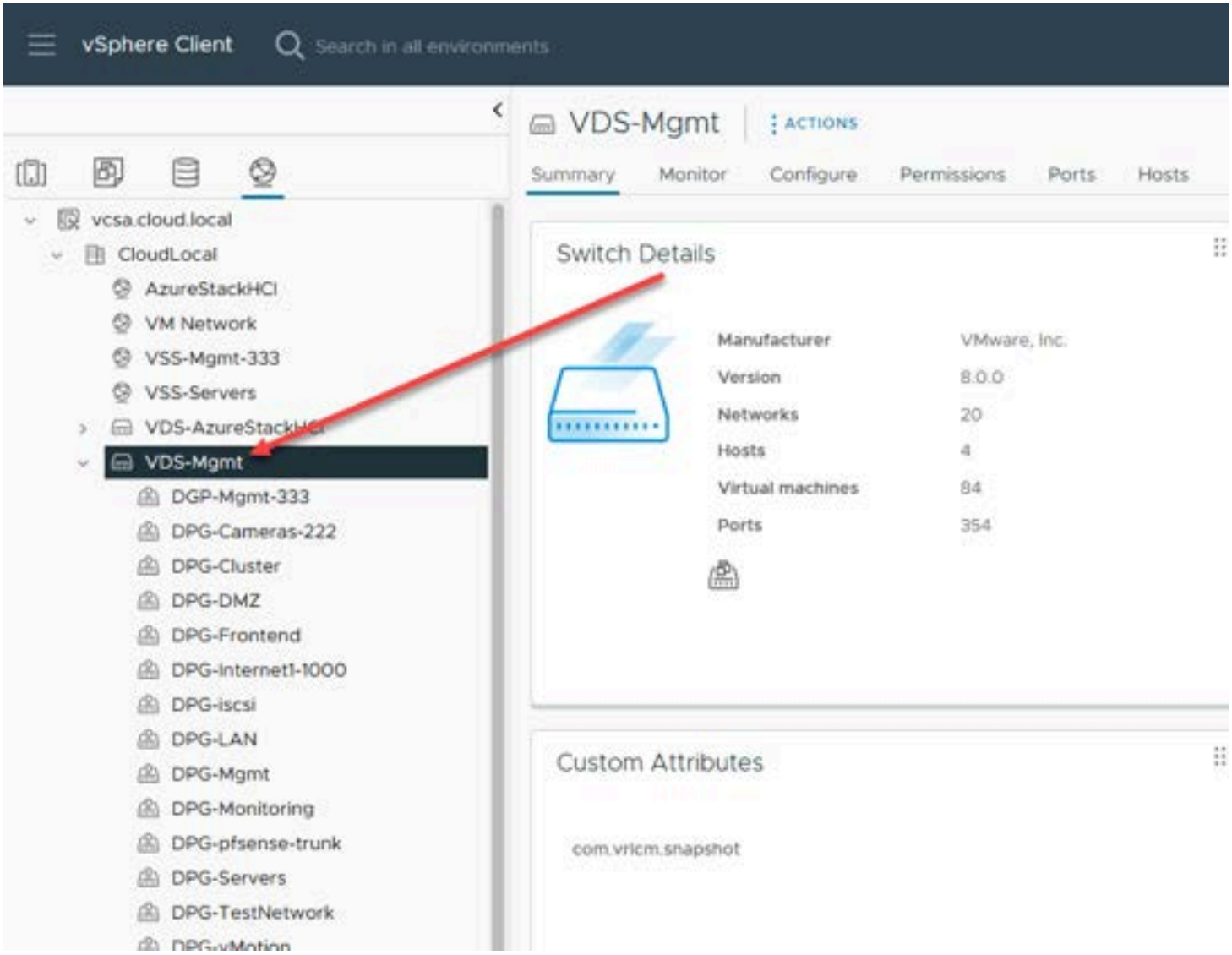
Now that we have created the child VLAN interface on the vmbr0 Linux bridge, you can see the **vmbr0.333** interface listed now under the network configuration in the navigation tree of **System > network**.



Viewing the newly created linux vlan

3. Distributed Switches

In case you are wondering, Proxmox doesn't have an equivalent construct like the vSphere Distributed Switch. The vSphere distributed switch is managed and controlled from vCenter Server. Proxmox doesn't have a centralized management platform like vCenter Server.



Vmware vsphere distributed switch

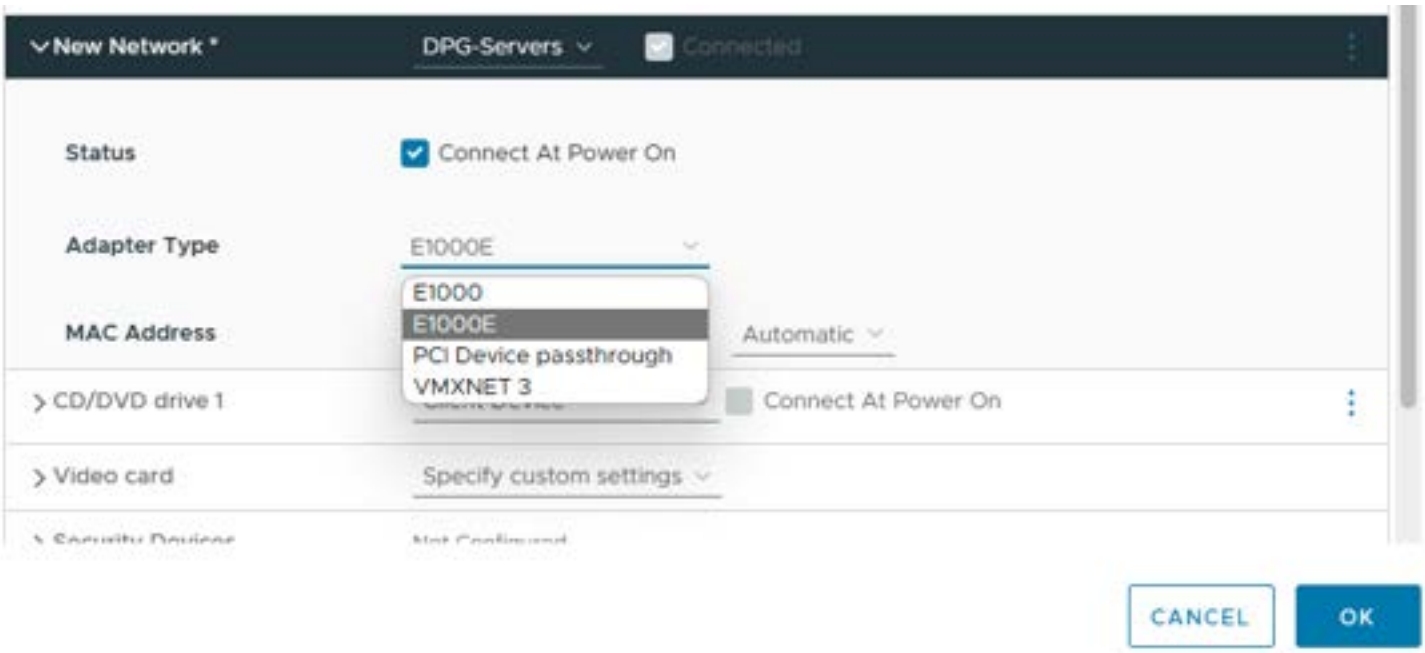
Proxmox admins would need to manage complex network setups manually with scripting or use third-party tools available for Proxmox for centralized network management.

4. Network Adapters

When it comes to network adapters for a virtual machine or container, both Proxmox and VMware vSphere support different types of network adapters. These include:

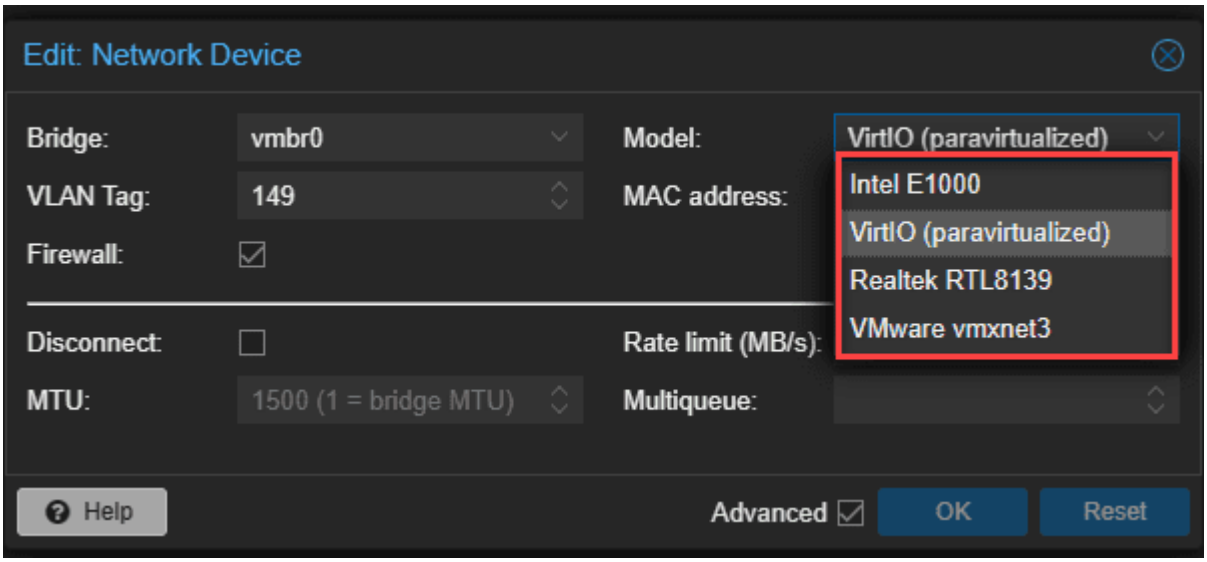
- VMXNET3
- E1000
- PCI Passthrough
- VirtIO (Proxmox)

Below you see the vnic options for a virtual machine.



Vmware network adapters

Below are the options when creating a new virtual machine in Proxmox.



Proxmox virtual network adapters

5. Network I/O Control

VMware has many tools for network traffic priority, bandwidth allocation, and other network capabilities. Proxmox does not contain network I/O control features built-in that are found in VMware vSphere. You can use Linux tools in Proxmox to help control network bandwidth.

With Proxmox, you can take advantage of one or many of the following Linux networking tools:

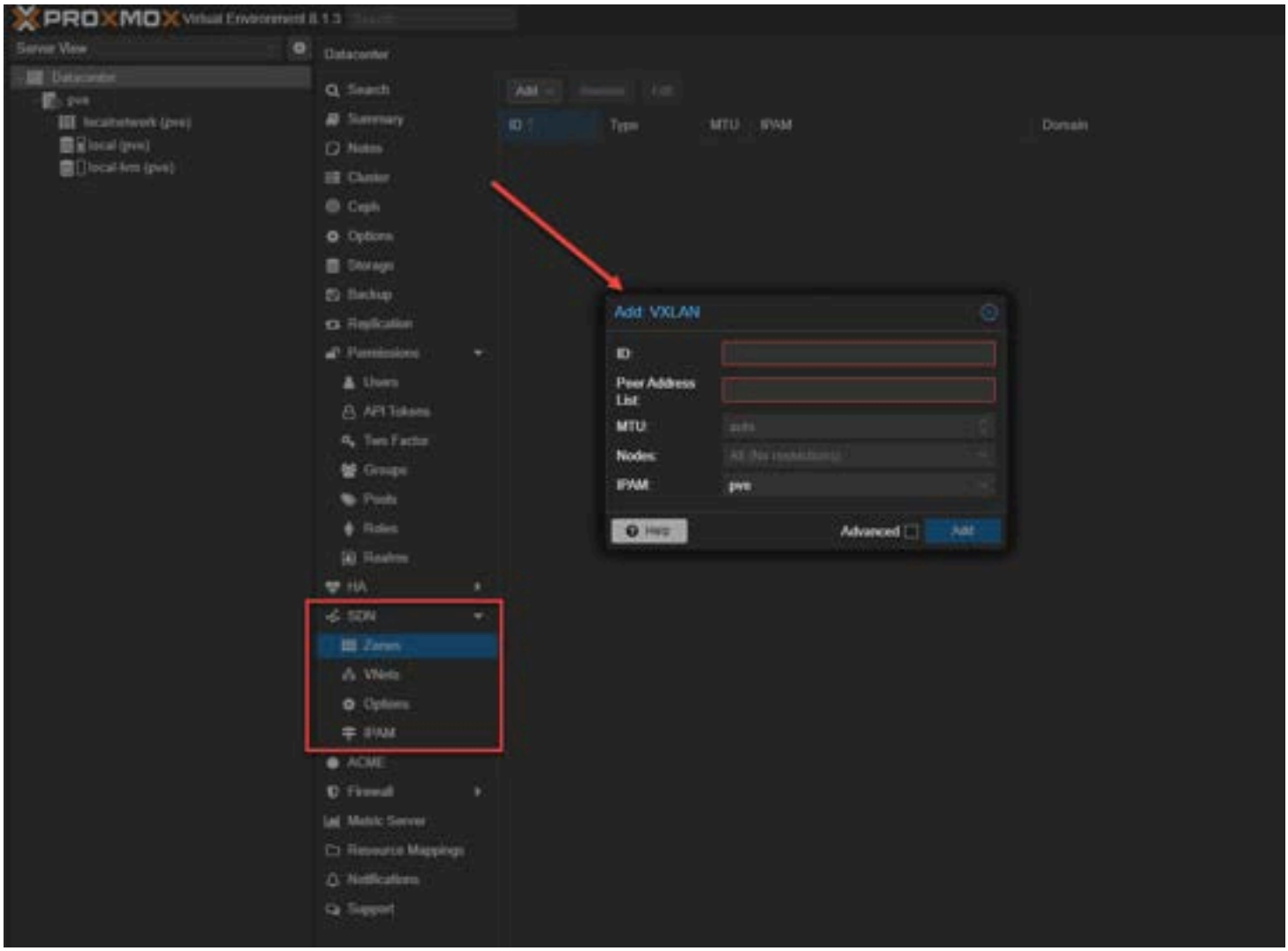
1. **Traffic Control (tc)**:
   - tc is a tool in the Linux iproute2 package. It allows you to control the traffic going through your network interfaces. You can create rules for traffic shaping and prioritization. You can also use it for Quality of Service (QoS) rules.
2. **iptables**:
   - iptables can also be used for basic network traffic control. It allows packet filtering and can be combined with tc for more granular control.

3. **ethtool**:
    - ethtool is used for querying and controlling network driver and hardware settings. It can be used to adjust settings like speed, duplex, and autonegotiation on Ethernet interfaces, which can indirectly influence network performance.
4. **nftables**:
    - nftables can be used to set up basic traffic control mechanisms.
5. **Wondershaper**:
    - Wondershaper is a simpler tool designed to limit the bandwidth of specific network interfaces. It's a good choice for basic bandwidth management.
6. **VLAN Configuration**:
    - Configuring VLANs like we discussed above, can help segment network traffic for more efficient network utilization. Linux's native VLAN configuration tools can be used with Proxmox for this purpose.
7. **Network Namespaces**:
    - Linux network namespaces can be used to isolate network environments for different VMs or containers. This can help manage network traffic and ensure that different services don't interfere with each other's network resources.
8. **Monitoring Tools (like iftop, nload, bmon)**:
    - While not directly involved in controlling traffic, monitoring tools are crucial for understanding network usage and identifying bottlenecks. Tools like iftop, nload, or bmon provide real-time network bandwidth usage information.
9. **Cgroups (Control Groups)**:
    - Cgroups, a Linux kernel feature, can be used to limit and prioritize network bandwidth usage among different processes and VMs. While more commonly used for managing CPU and memory resources, cgroups can also be configured to control network I/O.

6. Software-defined networking (SDN)

VMware has a very well-known platform for software-defined networking, known as VMware NSX. The NSX platform is a paid solution on top of vSphere that allows admins to create logical software-defined overlay networks on top of the physical underlay network.

New with Proxmox 8.1 is the introduction of software-defined networking capabilities. You can read the official documentation here. The latest version of Proxmox VE comes with core SDN packages pre-installed. You now have the option for SDN technology in Proxmox VE, allowing admins to create virtual zones and networks (VNets). SDN can also be used for advanced load balancing, NAT, and other features.



Software defined networking in proxmox 8.1

Admins can administer intricate network configurations and multi-tenant environments directly through the web interface at the datacenter level in Proxmox. It allows creating network infrastructure that is more adaptive and responsive and can scale in line with evolving business requirements.

7. Troubleshooting

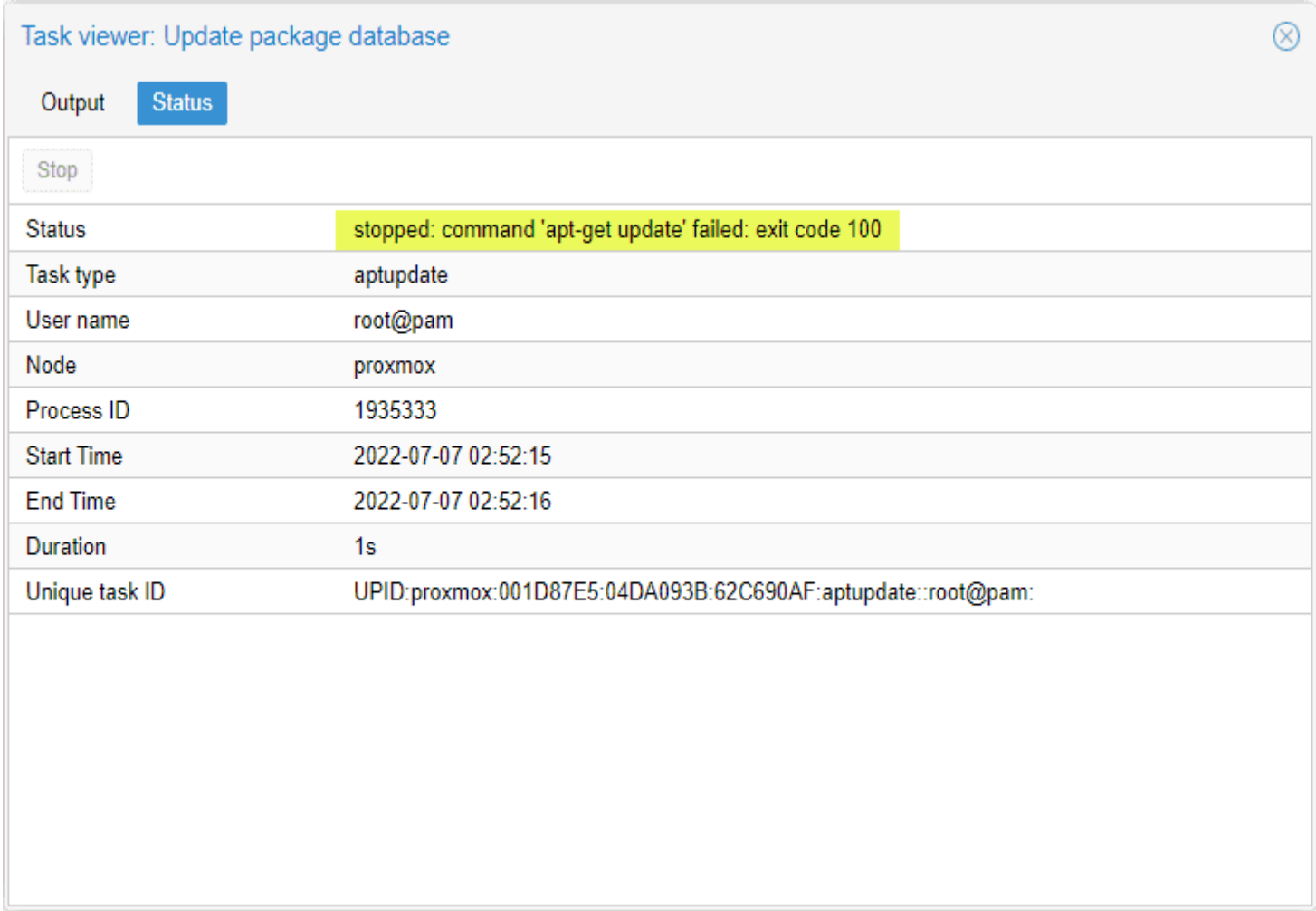
As you start to work with Proxmox networking, there may be a need for troubleshooting things when networking isn't working correctly. Checking the obvious things like VLANs, VLAN tagging configuration, both in Proxmox, and on your physical network switch are important. If you are using DHCP and DNS to connect to the host, is DHCP handing out the correct IP, and do you have records to resolve the Proxmox host?

Wrapping up

No doubt you have seen various posts and content thread posts from the search forums and community support forums like the Proxmox support forum related to networking issues. These can be challenging, especially when coming from another hypervisor. Hopefully, this post will help visitors understand Proxmox networking and the security enhancements available like VLANs, SDN, and others. Proxmox networking isn't so difficult to setup once you understand the equivalents from other virtualization environments you may be familiar with.

Proxmox Update No Subscription Repository Configuration

August 23, 2022
Proxmox



Update package database error

If you are delving into running Proxmox VE for your home lab or other use cases and are coming from other hypervisors you may have been playing around with, like me, you may struggle a bit with some of the basics when getting started learning the platform. One of those tasks is updating Proxmox to the latest Proxmox VE version. Let's take a look at how to update repositories and perform a dist upgrade to the latest version without having a Proxmox subscription.

Learn how to install Proxmox VE in VMware vSphere:

- Nested Proxmox VMware installation in ESXi

What is Proxmox VE?

Proxmox VE is an enterprise hypervisor that I have seen really gaining popularity among the home usage community and elsewhere as it provides a readily available and Proxmox works with most hardware that other hypervisors work with.

Proxmox VE is a complete open-source virtualization platform for enterprise virtualization. With PVE you can run virtual machines and even containers with your Proxmox VE installation.

It also includes a free Proxmox Backup Server that provides an enterprise backup solution for backing up and recovering your virtual machines, containers, and physical hosts, all in one solution.

Proxmox VE enterprise virtualization hypervisor

You can learn more about and download Proxmox VE from here:

- Proxmox – Powerful open-source server solutions

Why upgrade Proxmox VE?

Like any good lifecycle management, upgrading ProxMox VE is best practice. The last thing you want to do is neglect to upgrade your hypervisor platform that you are running your critical virtual machines and containers on. Performing a dist upgrade ensures getting the latest security and other updates for your Proxmox VE solution.

There are a couple of ways you upgrade your Proxmox VE installation, using the Proxmox web interface, or using the apt get update proxmox ve and apt get upgrade commands from the command line, either at the console or from an SSH connection.

Proxmox VE no subscription upgrade challenges

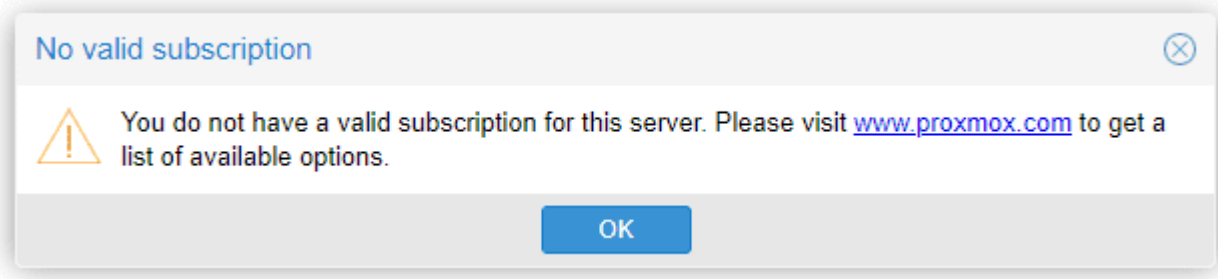
One of the challenges when starting off with Proxmox VE you may run into is Proxmox VE asks for a valid subscription to upgrade the platform. If you have a PVE no subscription installation, how do you perform a run disk upgrade on the hypervisor for non-enterprise use?

The good news is even if you have a non-licensed version, non-PVE enterprise installation that is not a paid version, you can still retrieve software upgrades on your non-enterprise version to update Proxmox.

Like all other Linux distributions, upgrades and updates pull from a repository. Proxmox VE by default is geared towards production use, and the update and upgrade repositories are pointed to the enterprise repository locations accordingly.

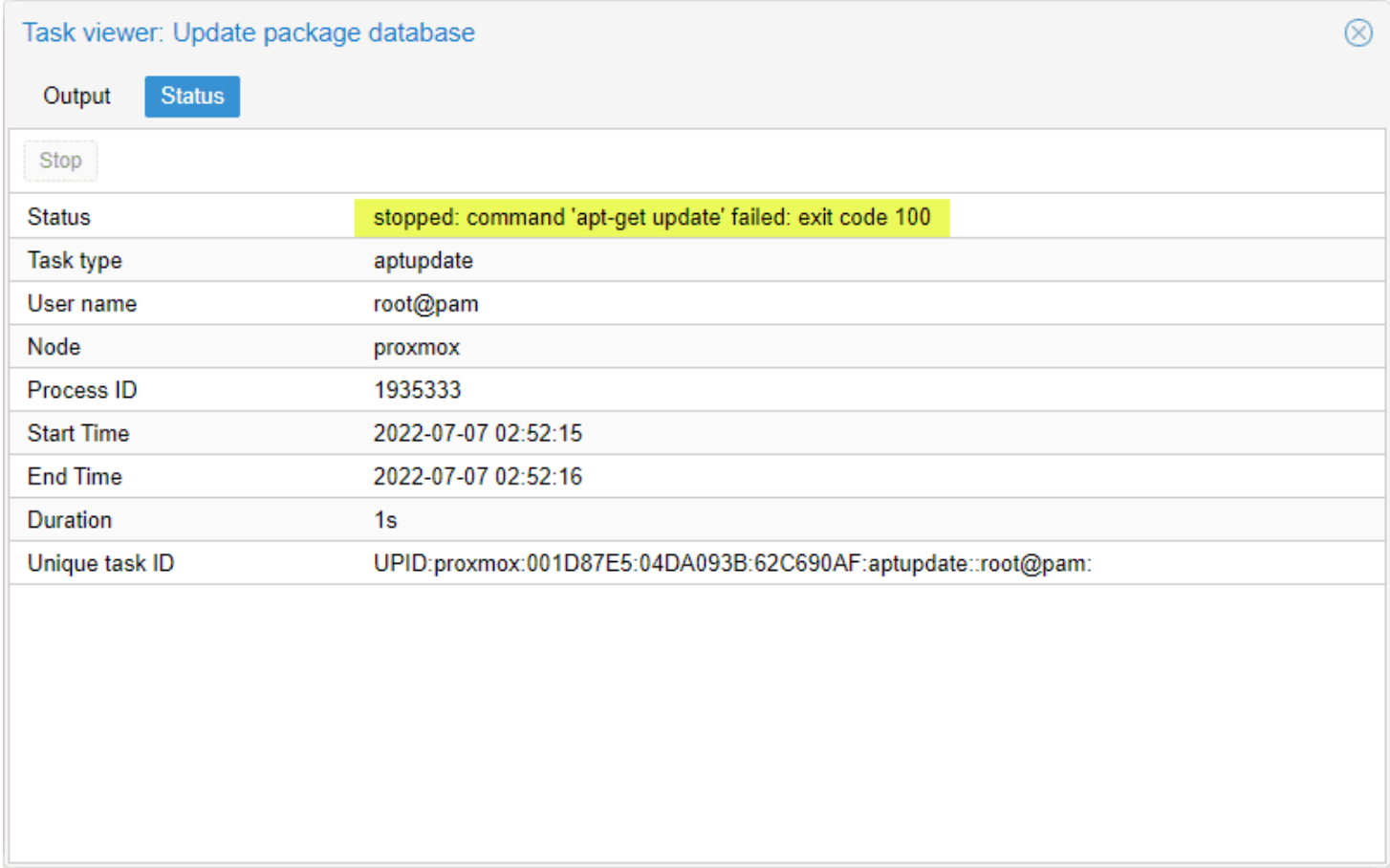
Default Proxmox VE upgrade settings

The default Proxmox VE upgrade settings point to enterprise repositories. So, when you run software upgrades using the run dist upgrade command you may see the error that you are running PVE no subscription.



No valid subscription configuration

This is because, by default, Proxmox VE points to the enterprise repo to pull down package lists. So, when you download and install Proxmox VE, it is set up for PVE enterprise and the PVE no subscription configuration is something you can introduce. Let's work on the PVE no subscription repository subscription repository.



Update package database error

Proxmox Update No Subscription Repository Configuration

What steps are needed to pivot from the enterprise repository to a no subscription configuration with Proxmox VE? The enterprise repository is defined in the etc apt sources.list.d configuration file. The PVE no subscription repository configuration is defined in the repository files.

Proxmox 8 and higher

The files changed a little with Proxmox 8 and higher. Note the following changes you need to make:

```
#/etc/apt/sources.list.d/pve-enterprise.list

From: deb https://enterprise.proxmox.com/debian/pve bookworm enterprise
To: deb http://download.proxmox.com/debian/pve bookworm pve-no-subscription

#/etc/apt/sources.list.d/ceph.list
#For Ceph Quincy
From: deb https://enterprise.proxmox.com/debian/ceph-quincy bookworm enterprise
To: deb http://download.proxmox.com/debian/ceph-quincy bookworm no-subscription

#For Ceph Reef
From: deb https://enterprise.proxmox.com/debian/ceph-reef bookworm enterprise
To: deb http://download.proxmox.com/debian/ceph-reef bookworm no-subscription
```

Before Proxmox 8

The steps for setting up a PVE no subscription configuration is configured using the etc apt sources.list.d file found at:

/etc/apt/sources.list

Add the following line in the /etc/apt/sources.list file:

deb http://download.proxmox.com/debian/pve bullseye pve-no-subscription

```
  GNU nano 5.4                          /etc/apt/sources.list
deb http://ftp.us.debian.org/debian bullseye main contrib

deb http://ftp.us.debian.org/debian bullseye-updates main contrib

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

# non production use updates
deb http://download.proxmox.com/debian/pve bullseye pve-no-subscription

                                        [ Read 9 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^_ Go To Line M-E Redo
```

Adding the pve no subscription line in the configuration file

Now, we just need to comment out a line in another file located here:

/etc/apt/sources.list.d/pve-enterprise.list

```
  GNU nano 5.4                    /etc/apt/sources.list.d/pve-enterprise.list
#deb https://enterprise.proxmox.com/debian/pve bullseye pve-enterprise

                                          [ Read 1 line ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^_ Go To Line M-E Redo
```

Editing the pve enterprise.list file

After editing and saving both of the above files, we need to run an **apt-get update** proxmox VE command at the command line.

```
root@proxmox:~# apt-get update
Hit:1 http://ftp.us.debian.org/debian bullseye InRelease
Hit:2 http://ftp.us.debian.org/debian bullseye-updates InRelease
Hit:3 http://security.debian.org bullseye-security InRelease
Hit:4 http://download.proxmox.com/debian/pve bullseye InRelease
Reading package lists... Done
root@proxmox:~#
```

Running an apt get update

After updating the repository with the non enterprise repo, we can perform a non pve enterprise repository upgrade using the command:

apt dist-upgrade

As you can see below, I have an upgrade that is available for the Proxmox VE server ready to install after configuring the upgrade to bypass the subscription requirement.

```
Hit:2 http://ftp.us.debian.org/debian bullseye-updates InRelease
Hit:3 http://security.debian.org bullseye-security InRelease
Hit:4 http://download.proxmox.com/debian/pve bullseye InRelease
Reading package lists... Done
root@proxmox:~# apt dist-upgrade
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
The following NEW packages will be installed:
  libdrm-common libdrm2 libepoxy0 libgbm1 libproxmox-rs-perl libvirglrenderer1 libwayland-server0
  proxmox-websocket-tunnel pve-kernel-5.13.19-6-pve pve-kernel-5.15 pve-kernel-5.15.39-4-pve
The following packages will be upgraded:
  base-files bash bind9-dnsutils bind9-host bind9-libs bsdextrautils bsdutils btrfs-progs chrony
  cifs-utils curl dirmngr distro-info-data dpkg e2fsprogs eject fdisk gnupg gnupg-l10n gnupg-utils
  gnutls-bin gpg gpg-agent gpg-wks-client gpg-wks-server gpgconf gpgsm gpgv gzip libarchive13
  libblkid1 libc-bin libc-l10n libc6 libcom-err2 libcryptsetup12 libcups2 libcurl3-gnutls libcurl4
  libexpat1 libext2fs2 libfdisk1 libflac8 libfreetype6 libfribidi0 libgmp10 libgnutls-dane0
  libgnutls30 libgnutlsxx28 libknet1 libldap-2.4-2 libldb2 liblzma5 libmount1 libnozzle1
  libnss-systemd libnss3 libnvpair3linux libpam-systemd libproxmox-acme-perl
  libproxmox-acme-plugins libproxmox-backup-qemu0 libpve-access-control libpve-cluster-api-perl
  libpve-cluster-perl libpve-common-perl libpve-guest-common-perl libpve-http-server-perl
  libpve-rs-perl libpve-storage-perl libpve-u2f-server-perl libsasl2-2 libsasl2-modules-db
  libseccomp2 libsmartcols1 libsmbclient libss2 libssl1.1 libsystemd0 libtirpc-common libtirpc3
  libtpms0 libudev1 libuuid1 libuutil3linux libwbclient0 libxml2 libzfs4linux libzpool5linux
  locales logrotate logsave lxc-pve lxcfs mount nano novnc-pve openssh-client openssh-server
  openssh-sftp-server openssl postfix procmail proxmox-backup-client proxmox-backup-file-restore
  proxmox-ve proxmox-widget-toolkit pve-cluster pve-container pve-docs pve-edk2-firmware
  pve-firmware pve-ha-manager pve-i18n pve-kernel-5.13 pve-kernel-helper pve-lxc-syscalld
  pve-manager pve-qemu-kvm pve-xtermjs python3-ldb qemu-server rsyslog samba-common samba-libs
  smartmontools smbclient spl ssh swtpm swtpm-libs swtpm-tools systemd systemd-sysv sysvinit-utils
  tasksel tasksel-data tcpdump tzdata udev util-linux vim-common vim-tiny wget xxd xz-utils
  zfs-initramfs zfs-zed zfsutils-linux zlib1g
150 upgraded, 11 newly installed, 0 to remove and 0 not upgraded.
Need to get 336 MB of archives.
After this operation, 786 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Running an apt dist upgrade command from the command line

Proxmox VE upgrade FAQs

What is Proxmox VE?

Proxmox VE is an open-source server management platform for enterprise virtualization. It provides integration with the KVM hypervisor and Linux Containers (LXC), software-defined storage and networking functionality, on a single platform. You can use the web-based user interface to manage virtual machines, LXC containers, Proxmox clusters, or integrate disaster recovery tools.

Why am I getting a Proxmox subscription error about updates?

By default, Proxmox VE is pointed to the enterprise repositories which requires a subscription to perform updates. However, this is a minor configuration change to bypass the enterprise repo and point to the non enterprise repo for pulling down updates.

How do I configure Proxmox for the non enterprise repository?

There are essentially two files that you need to edit, the **/etc/apt/sources.list** file and the **/etc/apt/sources.list.d/pve-enterprise.list** file. After editing the files with the configuration listed above, you run an **apt-get update** and then the command **apt dist-upgrade**.

Is Proxmox free?

Yes, Proxmox is free to download and install in your environment. Additionally, as shown, you can change from the enterprise version of the update proxmox repository to the non enterprise version.

Wrapping Up

Proxmox VE is a great platform for the home lab or for enterprise use and provides many great capabilities to run virtual machines and containerized workloads in your environment. By editing just a few minor configuration files, you can easily bypass the requirement for the subscription when updating Proxmox VE installations with the latest upgrades. It allows keeping Proxmox installations up to date with the latest security patches and other upgrades from Proxmox.

Proxmox VLAN Configuration: Management IP, Bridge, and Virtual Machines

December 11, 2023
Proxmox

Proxmox vlans

Proxmox is a free and open-source hypervisor with enterprise features for virtualization. Many may struggle with Proxmox networking and understanding concepts such as Proxmox VLAN configuration. If you are running VLANs in your network, you may want your Proxmox VE management IP on your management VLAN, or you may need to connect your virtual machines to separate VLANs. Let's look and see how we can do this.

Table of contents

What are VLANs?

First, let's get a quick primer on what VLANs are exactly. VLANs (**virtual local area networks**) are logical networks you can create that separate devices and broadcast domains. They enable isolating network traffic between devices all without having separate physical network infrastructure. VLAN-aware network switches can assign a VLAN tag that identifies a unique network and broadcast domain.

Network and VLAN terms

Before diving into VLANs, let's review some essential networking concepts:

network device: A network device is really anything (physical or virtual) that can connect to a computer network

Linux Bridge: A Linux bridge enables more than one network interface to act as a single network device.

Virtual Machine (VM): A virtualized instance of an operating system running on a hypervisor

IP Address: A numeric identifier of network devices on a network. These must be unique.

Default Configuration: The initial settings applied to a device or software.

Networking Service: Software that manages network connections and traffic flow

Management Interface: In Proxmox VE this is the network interface that allows you to access the web UI and command line interface of your Proxmox host.

Physical Network Interface (NIC): The physical connection from a computer to a physical network switch port.

Network Interfaces File: In Linux systems this is where you setup the network configuration for your network interfaces.

Proxmox default network configuration

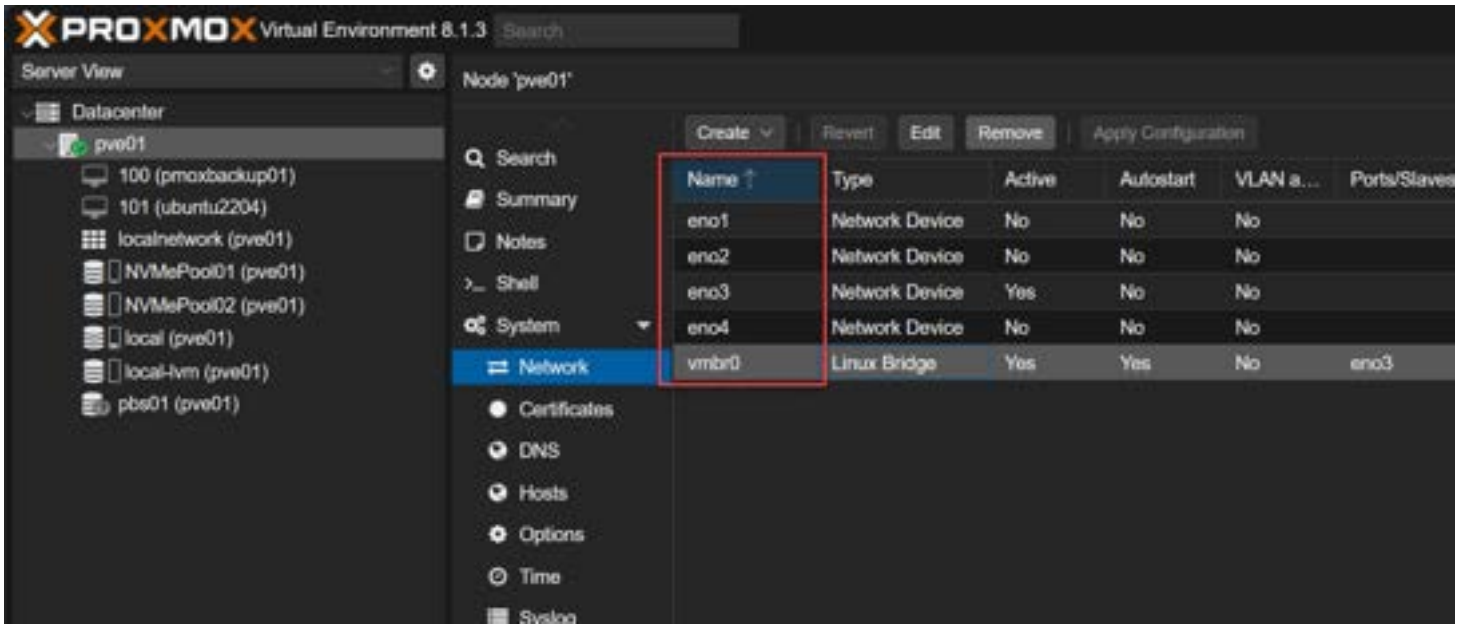
In the below screenshots, I am using one of my Supermicro hosts that is configured with (2) 1 GbE connections and (2) 10 GbE connections.

In the Proxmox network connections, you will see the individual physical adapters and then you will see the Proxmox Linux bridge configured by default.

Proxmox ve 1

Below:

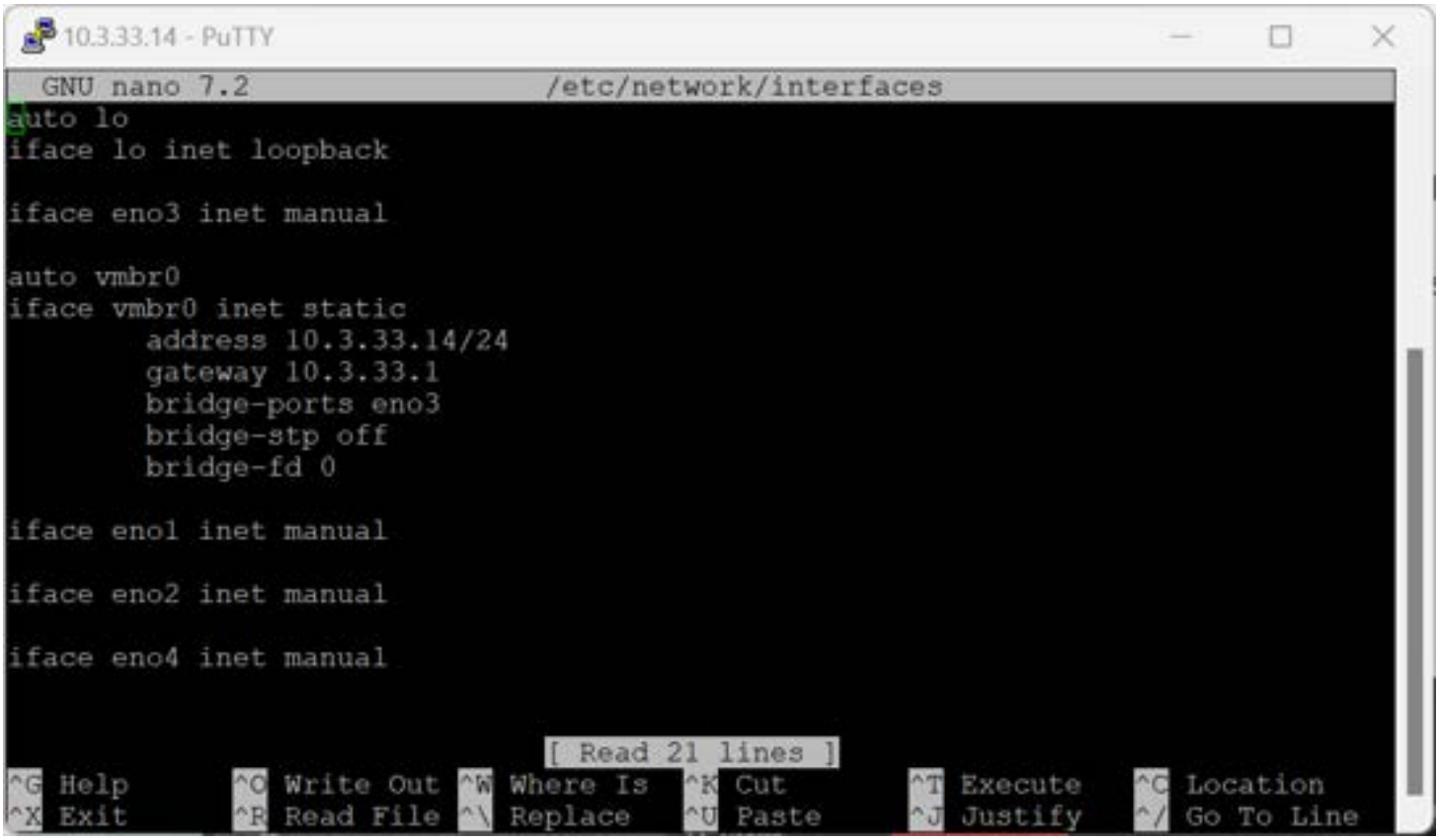
- Individual physical adapters are named **eno1, eno2, eno3, eno4**
- The Linux bridge is called **vmbr0**



Viewing the default promxox network configuration after installation

You can look at the low-level configuration in the following file:

`/etc/network/interfaces`



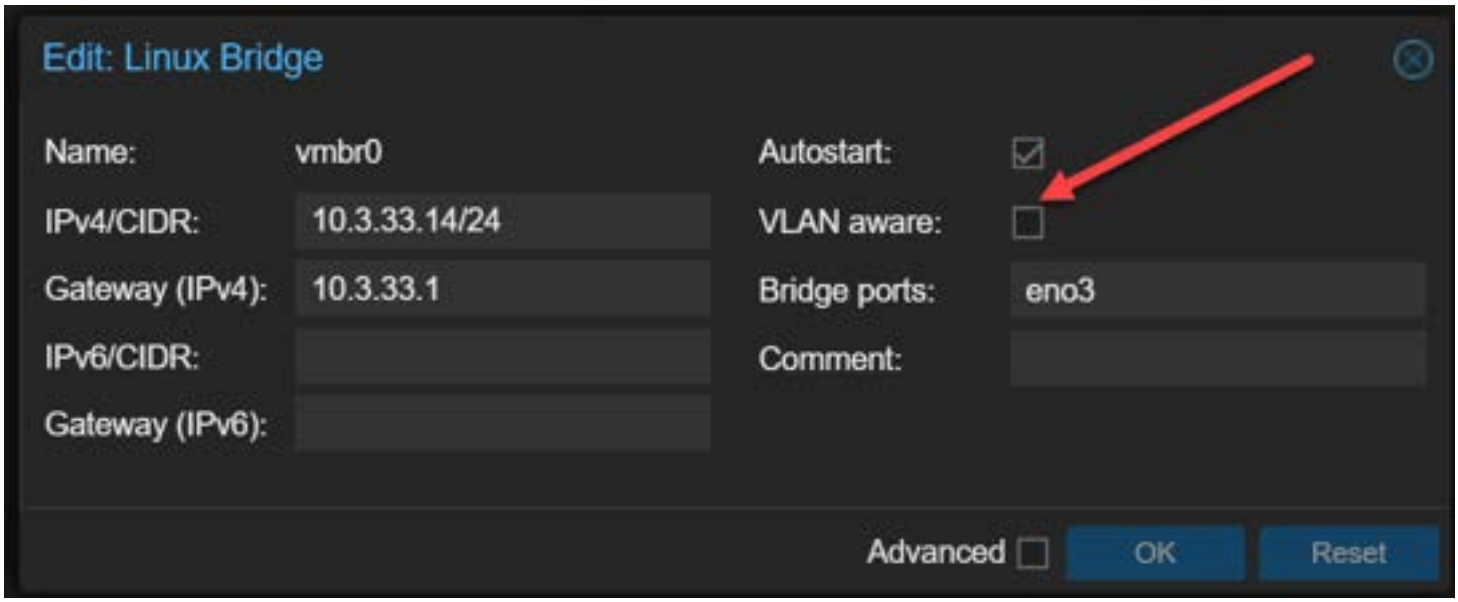
Viewing the network configuration from the network interfaces file

Make the default Proxmox VE Linux bridge VLAN-aware

One of the easiest configurations to implement Proxmox VLANs is called **bridge VLAN aware**. With this configuration, you are simply enabling VLANs on the default **vmbr0** interface.

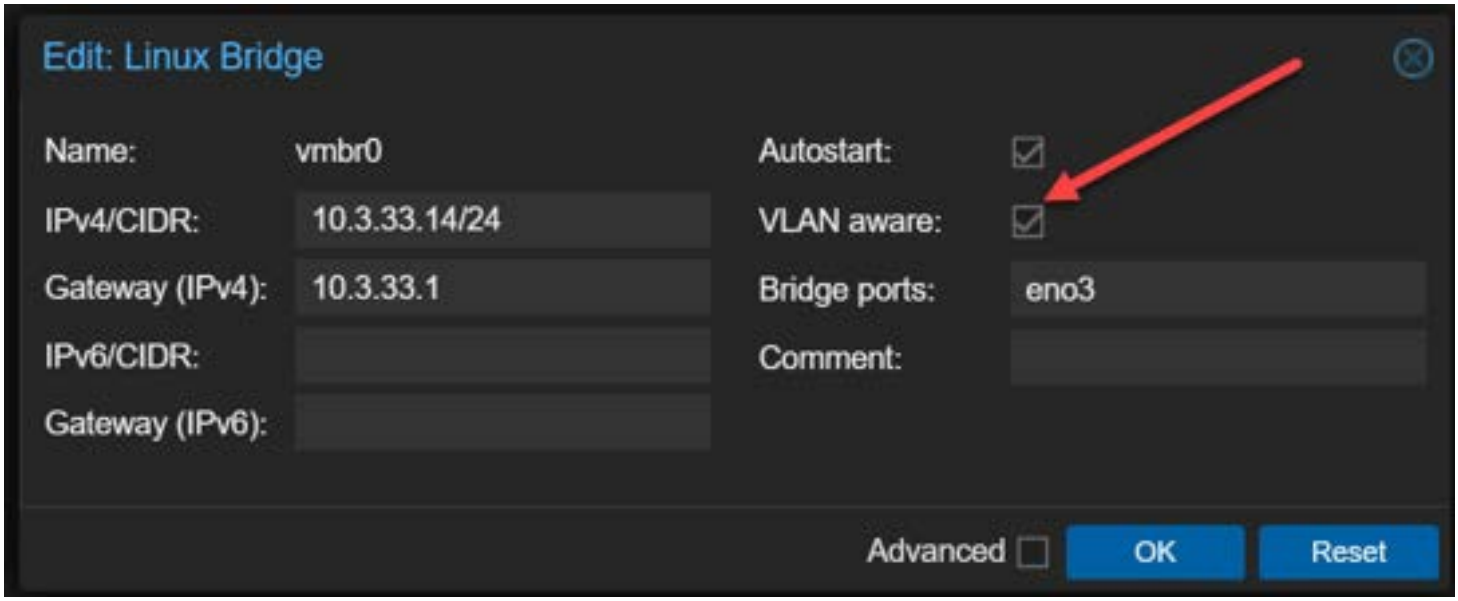
To do this, open the properties of the **vmbr0** interface under your proxmox host properties **Network > vmbr0 > Edit**.

You will see this by default. The **VLAN aware** setting will be unchecked. The bridge port is assigned with the interface that is uplinked.



Vlan aware check box

Now, to make our bridge VLAN-aware, place a check in the VLAN aware box. Click **OK**.

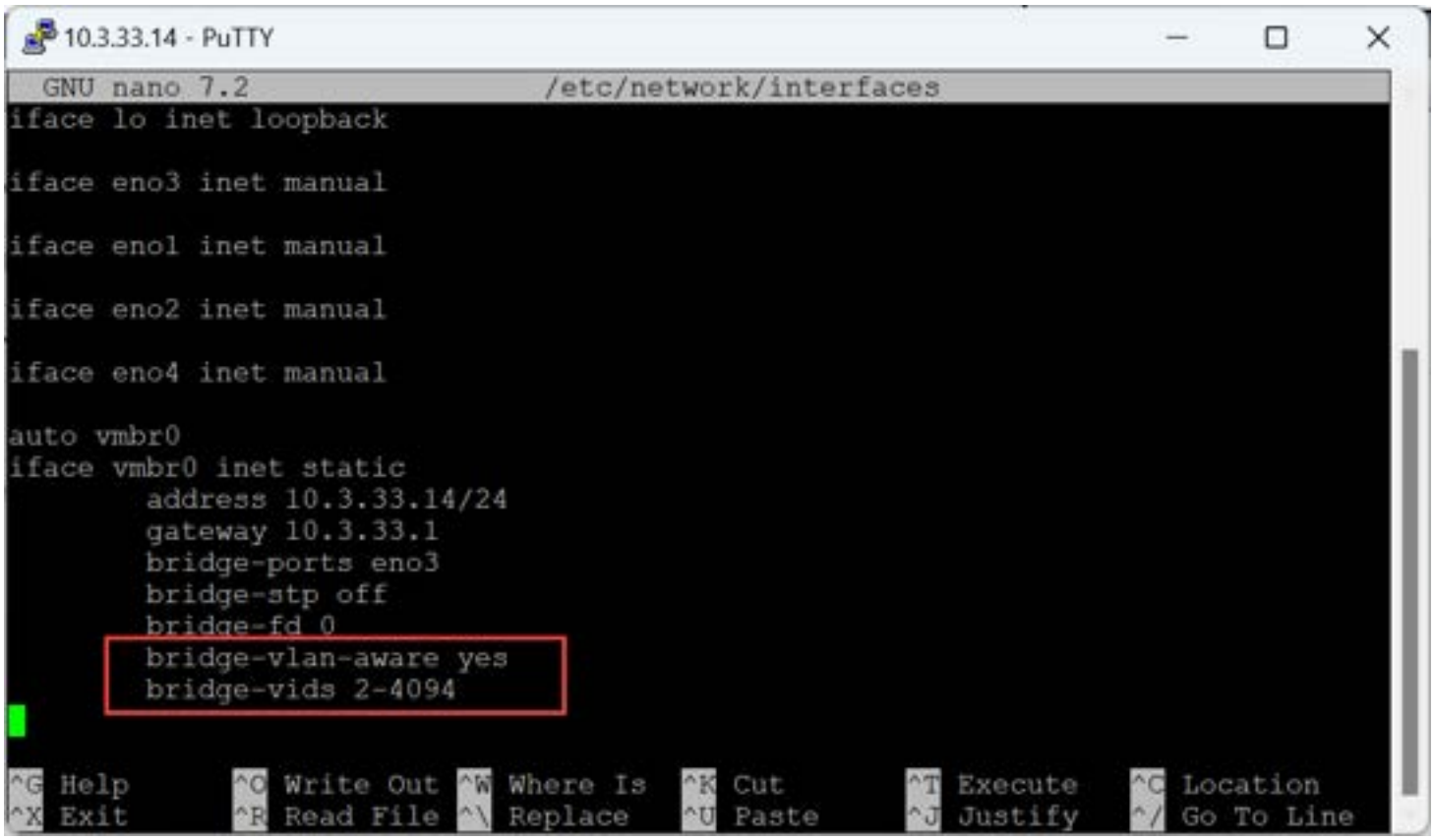


Enabling vlan aware on the default bridge

After you make the change, reboot your Proxmox VE host:

```
reboot
```

How does this change the **/etc/network/interfaces** file?



Configuration added to the network interfaces file after making vlan aware

You will see the configuration change and add the VLAN stanzas in the configuration, as you can see in my configuration.

```
iface lo inet loopback

iface eno3 inet manual

iface eno1 inet manual

iface eno2 inet manual

iface eno4 inet manual

auto vmbr0
iface vmbr0 inet static
address 10.3.33.14/24
gateway 10.3.33.1
bridge-ports eno3
bridge-stp off
bridge-fd 0
bridge-vlan-aware yes
bridge-vids 2-4094
```

```
auto vmbr0
iface vmbr0 inet manual
        address 10.3.33.14/24
        gateway 10.3.33.1
        bridge-ports eno3
        bridge-stp off
        bridge-fd 0
        bridge-vlan-aware yes
        bridge-vids 2-4094
```

Trunking configuration allowing multiple vlans

By default, Proxmox will enable the Linux bridge with a “trunk port” configuration that accepts all VLANs from 2-4094. You can remove all the VLANs aside from specific VLANs you want to tag, using the following configuration:

```
auto vmbr0
iface vmbr0 inet static
address 10.3.33.14/24
gateway 10.3.33.1
bridge-ports eno3
bridge-stp off
bridge-fd 0
bridge-vlan-aware yes
bridge-vids 10,149,222
```

Below, I have removed the IP from the default bridge, but you can see the restricted bridge-VIDs to specific VLANs.

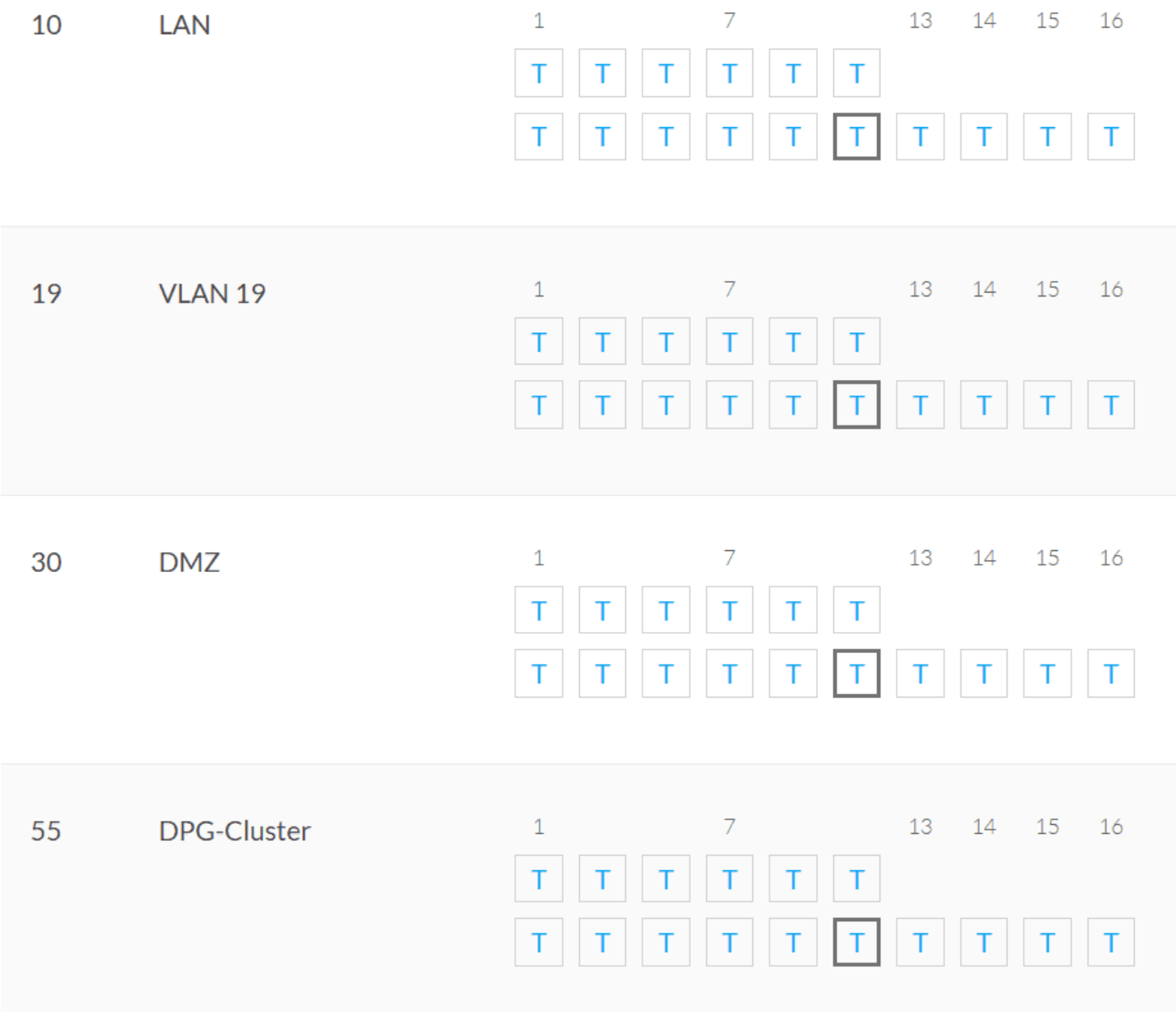
```
auto vmbr0
iface vmbr0 inet manual
        bridge-ports eno3
        bridge-stp off
        bridge-fd 0
        bridge-vlan-aware yes
        bridge-vids 10,149,222
```

Restricting vlans on the default bridge

Physical network switch tagging

One point to note at this point is that you need to make sure your physical network switch plinking your Proxmox host is tagged for all the VLANs you want the Proxmox bridge to communicate with. If we are tagging frames from the Proxmox side with VLAN IDs that the physical network switch does not have configured, the frames will be discarded.

Below is a screenshot of VLANs configuration and VLAN setup on my Ubiquiti 10 GbE switch. You can see the VLANs tagging and trunking configured on the switch. The **T** stands for “tagged”. As you can see below, I have VLANs 10, 19, and 30 tagged on all ports.



Viewing tagged interfaces on a physical network switch

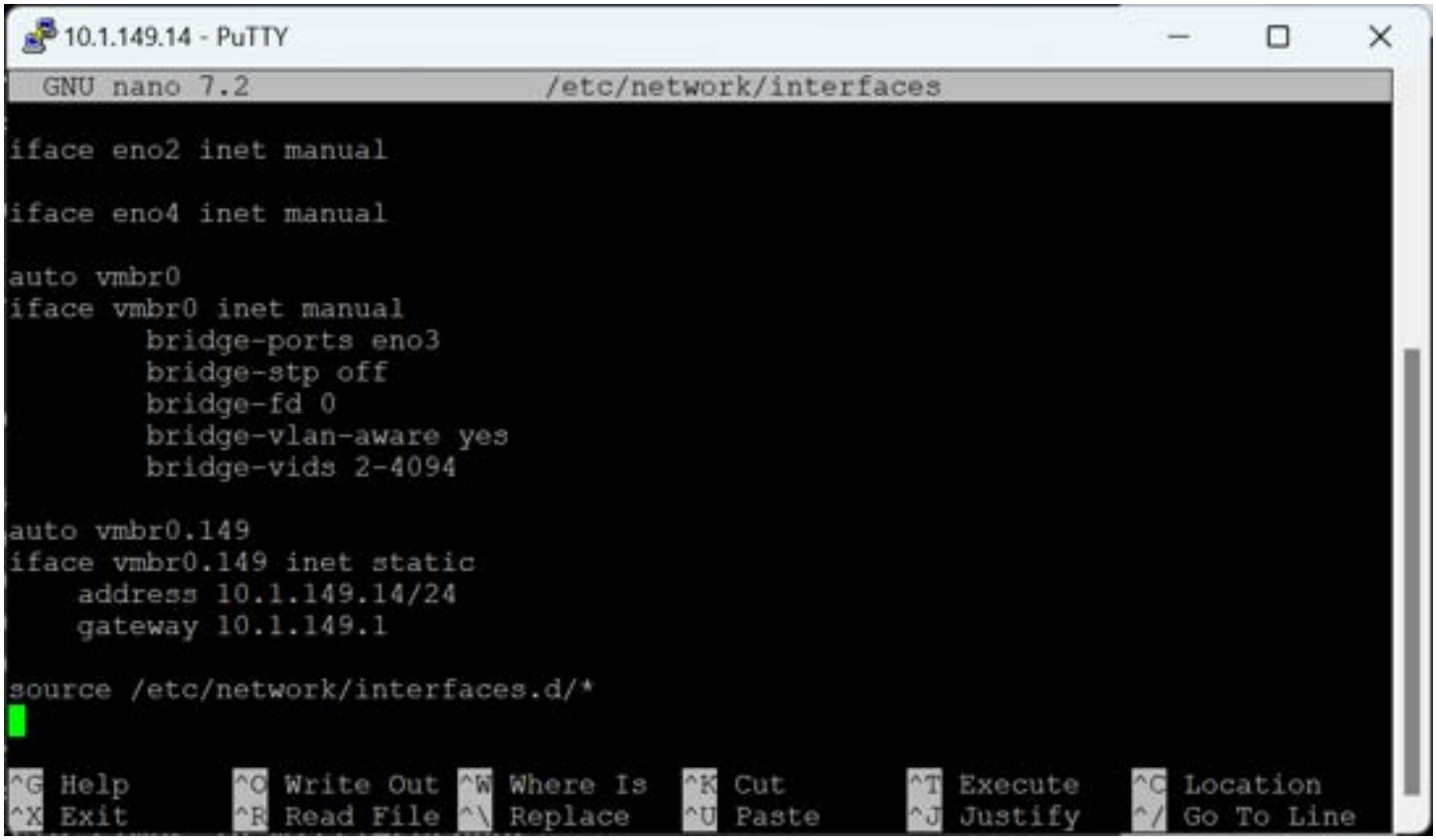
Setting the Proxmox Management interface IP on a different VLAN

What if we want to change the management interface IP and set the management interface IP on a different VLAN? We can do that with the following configuration. As we can see, I have removed the **address** and **gateway** configuration lines from the **vmbr0** configuration.

Instead, I have created a VLAN tagged interface, tagged with VLAN 149 for the management interface.

```
iface eno3 inet manual

iface eno1 inet manual

iface eno2 inet manual

iface eno4 inet manual

auto vmbr0
iface vmbr0 inet manual
```

```
bridge-ports eno3
bridge-stp off
bridge-fd 0
bridge-vlan-aware yes
bridge-vids 2-4094

auto vmbr0.149
iface vmbr0.149 inet static
address 10.1.149.14/24
gateway 10.1.149.1
```



Management ip for proxmox ve host on a different vlan

Save your configuration. You can reboot to make the configuration take effect, or you can run the command:ifup -a

Once you have rebooted or ran the ifup command, you should be able to run the **ip address** command to see the IP address and interfaces:ip a

```
10.1.149.14 - PuTTY
 1000
    link/ether ac:1f:6b:6c:1c:dd brd ff:ff:ff:ff:ff:ff
    altname enp8s0f1
4: eno3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master vmbr0 state
UP group default qlen 1000
    link/ether                          brd ff:ff:ff:ff:ff:ff
    altname enp3s0f0
5: eno4: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen
 1000
    link/ether                          brd ff:ff:ff:ff:ff:ff
    altname enp3s0f1
6: vmbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP grou
p default qlen 1000
    link/ether                          brd ff:ff:ff:ff:ff:ff
    inet6 fe80::ae1f:6bff:fe6c:222c/64 scope link
       valid_lft forever preferred_lft forever
7: vmbr0.149@vmbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue sta
te UP group default qlen 1000
    link/ether                          brd ff:ff:ff:ff:ff:ff
    inet 10.1.149.14/24 scope global vmbr0.149
       valid_lft forever preferred_lft forever
    inet6 fe80::ae1f:6bff:fe6c:222c/64 scope link
       valid_lft forever preferred_lft forever
root@pve01:~#
```

Viewing the ip a command to verify the ip address

We can also verify the configuration in the Proxmox VE GUI, looking at the properties of the Proxmox host **> Network**.

Create | Revert | Edit | Remove | Apply Configuration

Name ↑	Type	Active	Autostart	VLAN a...	Ports/Slaves	B
eno1	Network Device	No	No	No		
eno2	Network Device	No	No	No		
eno3	Network Device	Yes	No	No		
eno4	Network Device	No	No	No		
vmbr0	Linux Bridge	Yes	Yes	Yes	eno3	
vmbr0.149	Linux VLAN	Yes	Yes	No		

Viewing the new linux vlan from the proxmox gui

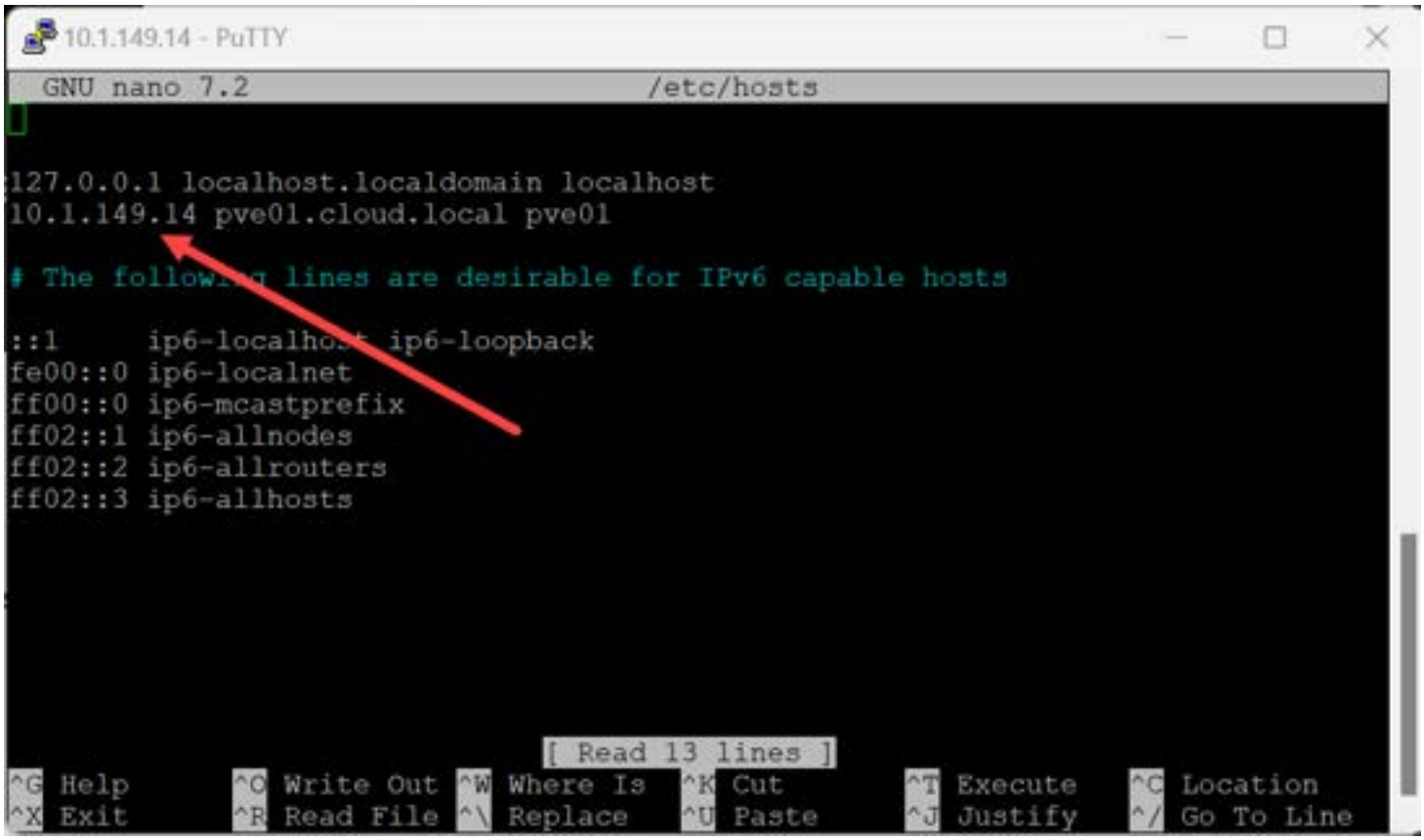
We can also check external connectivity with a simple ping of the new management IP address we have placed on the new VLAN.

```
Pinging 10.1.149.14 with 32 bytes of data:
Reply from 10.1.149.14: bytes=32 time=3ms TTL=63
Reply from 10.1.149.14: bytes=32 time=2ms TTL=63
Reply from 10.1.149.14: bytes=32 time=2ms TTL=63

Ping statistics for 10.1.149.14:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 3ms, Average = 2ms
```

Pinging the new management ip on the proxmox host

Change Proxmox VE host file reference to old IP

If you change your Proxmox VE management IP address, you will want to go into your **/etc/hosts** file and change the IP reference for the Proxmox host.nano /etc/hosts



Changing the linux hosts file

Configuring VLANs in Proxmox VE web interface

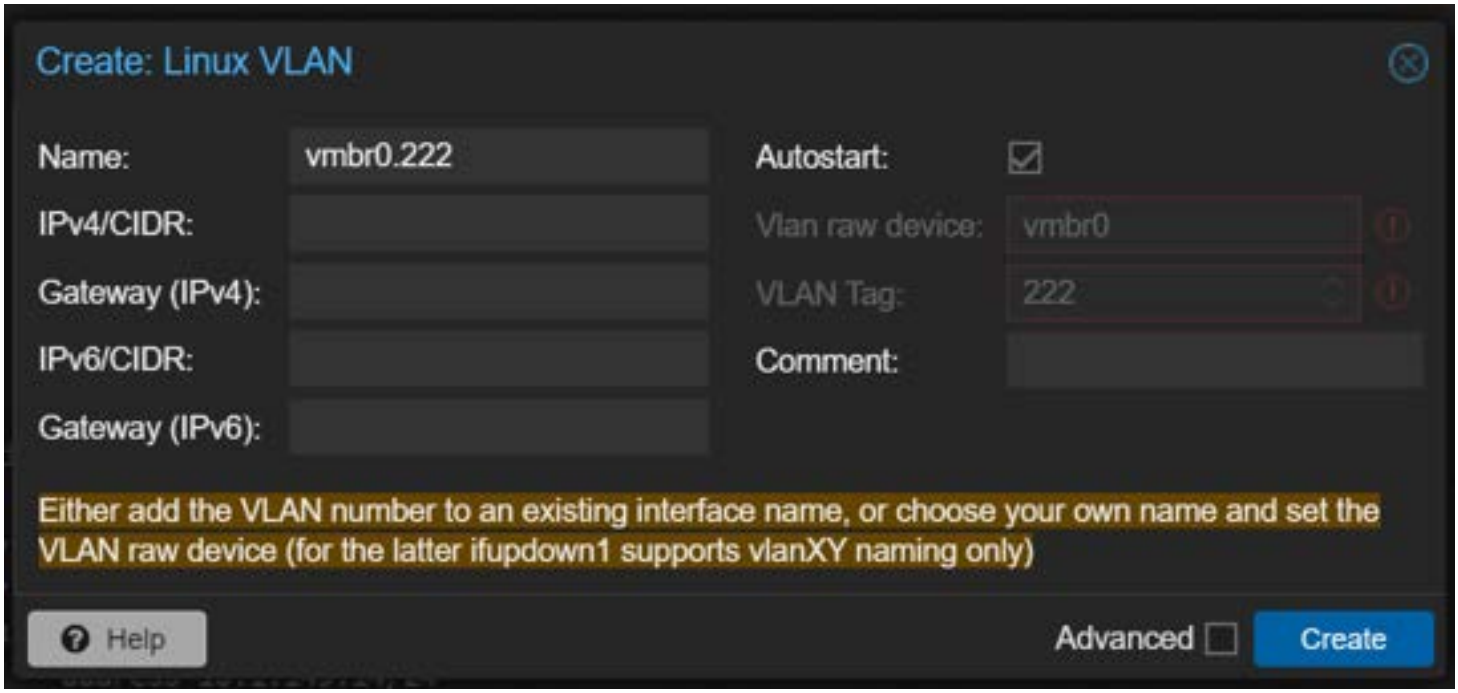
To configure VLANs in Proxmox VE on the default bridge using the web interface, we can follow the below.

Web Interface Configuration

The Proxmox VE web interface simplifies VLAN configuration through its GUI.

Create a new Linux VLAN:

- Go to the **Network** section in the web interface.
- Click on **Create** and select **Linux VLAN**.
- Enter a **VLAN ID** and a **Name**.
- Optionally, you can configure other settings, such as the bridge and the VLAN tag.
- Click on **Create** to save the changes.



Adding a new linux vlan from the proxmox gui

Advanced Configurations

Now that you understand the basics of VLAN configuration in Proxmox VE, we can explore some advanced topics:

Trunk Ports

A trunk port is a network interface that can carries multiple VLANs traffic. It is a useful configuration for connecting multiple VLANs and VMs to multiple VLANs. To configure a trunk port on Proxmox VE, you need to:

- Make the bridge VLAN aware
- Add the VLAN ID to the bridge configuration. By default it will be a trunking configuration when you make it VLAN aware. Proxmox automatically configures VLAN 2-4094 on the default bridge.
- Configure the VM network interface with the right VLAN tag

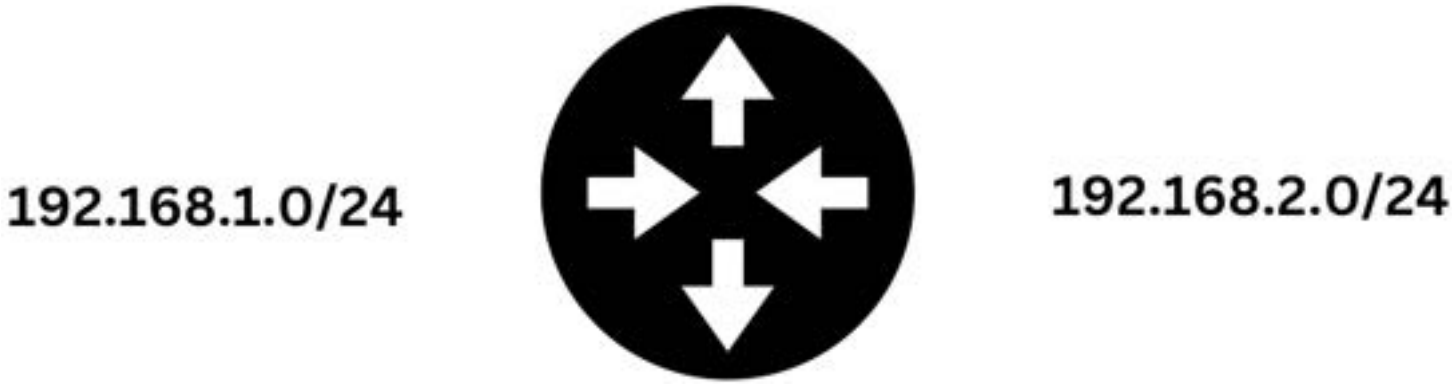
VLAN Aware Bridges

A VLAN aware bridge is a bridge that understands VLAN tags and can forward traffic to the correct VLAN. This is required for communicating between VMs on different VLANs. To configure a VLAN-aware bridge on Proxmox VE, you need to:

- Enable the vlan_filtering option in the bridge configuration.
- Add the VLAN ID to the bridge configuration.

Routing between VLANs

When you setup new VLANs, devices on one VLAN can't talk to the devices on the other subnet by default. Generally, according to best practice, a VLAN will house 1 subnet. So it means your devices on each VLAN will have different IP addresses on different subnets. You will need to configure a router or firewall that can do routing (like pfSense) between the devices on different VLANs/subnets so these can communicate.



Routing traffic between vlans

Troubleshooting

What if you have issues with your Proxmox VLANs?

- Check the syntax you have used in the **/etc/network/interfaces** file
- Make sure there isn't a mismatch between the VLAN tagging in Proxmox and the untagged VLAN on the Switch port. This could result in double-tagging frames (meaning your host is tagging a VLAN, and the Switch is also trying to tag it via untagged traffic)
- If you are restricting specific VLANs in your bridge VIDS configuration, make sure you have allowed the VLANs you are expecting to be tagged
- Make sure your network switch is tagged with all the VLANs on the physical uplink(s) for your Proxmox VE host
- If you are having trouble with one VLAN subnet talking to another VLAN subnet, make sure the appropriate routes are in place to make this happen

Frequently Asked Questions on Proxmox VLAN Configuration

How does VLAN tagging in Proxmox enhance network efficiency?

VLAN tagging helps to segment network traffic, which is key for managing network resources and maintaining traffic flow. This segmentation allows for better control and isolation of traffic, which is important in environments with multiple virtual machines.

What are the steps to configure a Linux bridge for VLANs in Proxmox?

Configuring a Linux bridge involves configuring the bridge interface in the network configuration file, setting the bridge to VLAN-aware, and assigning bridge ports.

Can VLANs improve security in a virtualized environment?

Absolutely. VLANs provide network isolation, a significant aspect of securing a virtualized environment. By segregating network traffic, VLANs help minimize the risk of unauthorized access or data breaches.

Why is setting the Proxmox VE management IP important in VLAN configuration?

The Proxmox VE management IP is used for remote management and access. VLAN configurations ensure that management traffic is isolated and secure, which is critical for security.

What are the challenges in configuring VLANs on a Proxmox VE host?

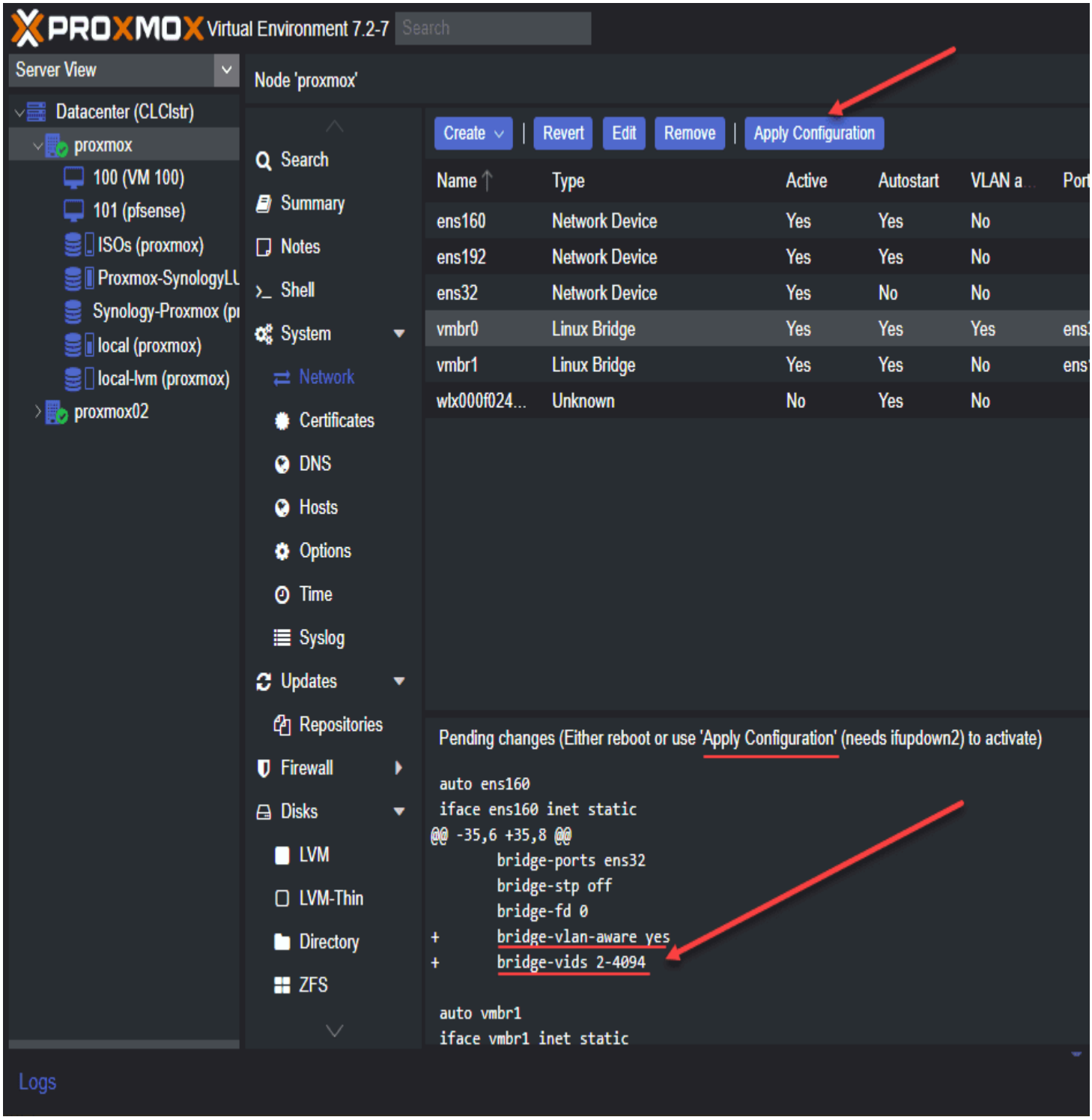
Misconfigurations can lead to disconnected hosts, VMs that can’t communicate correctly, and other communication issues. It is important to understand VLANs from the Proxmox VE host side and the physical network. Routing concepts also come into play to allow devices to communicate between VLANs.

Wrapping up

Creating and configuring VLANs in Proxmox is not too difficult. Once you understand the concepts and where to implement the configuration, it is actually quite simple. Adding VLANs to your Proxmox VE host will allow you to connect your virtualized workloads to the various networks that may be running in your network environment and enable traffic to flow and connect as expected.

Proxmox Management Interface VLAN tagging configuration

September 19, 2022
Proxmox



f588a257 2be7 4dec bb19 dc921a851d8b

If you have configured your Promox server in the home lab, most want to segregate their management traffic from the other types of traffic in their lab environment as part of their network configuration. Making the management interface VLAN aware ensures your Proxmox server can be connected to a trunk port and carry traffic for various VLANs. Let's see how to set up the Proxmox management interface VLAN tagging configuration and the steps involved.

Why segment your Promox VE management traffic?

First, why do you want to segment Proxmox VE management VLAN traffic from the rest of the traffic? Having management traffic on the same VLAN interface as virtual machines and other types of traffic is a security risk.

You never want to be able to manage the hypervisor host on the same network on which other clients and servers exist. as you can imagine, if an attacker has compromised the network where a client resides, you don't want them to have easy Layer 2 access to the management interface of your hypervisor.

What are VLANs?

VLAN traffic refers to "virtual local area network" traffic that essentially allows creating of many virtual networks on the same Ethernet wire. It is a layer 2 construct. Multiple VLANs allow segmenting traffic across the same physical network switch. Physical interfaces on switch ports are configured with VLAN-aware trunk port configuration, allowing the switch to see the VLAN tags added to Ethernet frames. You can essentially have one port that carries all the different VLAN networks. Two VLANs will flow across the same physical cabling and port.

There are many types of VLAN configurations. You can configure "untagged traffic," meaning traffic that does not have VLAN tagging, automatically get a specific VLAN tag. Generally, for many, the default VLAN is used for untagged traffic.

VLAN tagging can happen at the switch port level, or the network interface level, as well as the network interface tags VLAN traffic as it traverses the network. We can tag VLANs from the Proxmox side of things so that traffic is correctly tagged with the appropriate VLAN.

VLAN for Guest networks

Below is straight from the Proxmox documentation from the Proxmox server when you click the Linux vlan help button:

Proxmox VE supports this setup out of the box. You can specify the VLAN tag when you create a VM. The VLAN tag is part of the guest network configuration. The networking layer supports different modes to implement VLANs, depending on the bridge configuration:

VLAN awareness on the Linux bridge: In this case, each guest's virtual network card is assigned to a VLAN tag, which is transparently supported by the Linux bridge. Trunk mode is also possible, but that makes configuration in the guest necessary.

"traditional" VLAN on the Linux bridge: In contrast to the VLAN awareness method, this method is not transparent and creates a VLAN device with associated bridge for each VLAN. That is, creating a guest on VLAN 5 for example, would create two interfaces eno1.5 and vmbr0v5, which would remain until a reboot occurs.

Open vSwitch VLAN: This mode uses the OVS VLAN feature.

Guest configured VLAN: VLANs are assigned inside the guest. In this case, the setup is completely done inside the guest and can not be influenced from the outside. The benefit is that you can use more than one VLAN on a single virtual NIC.

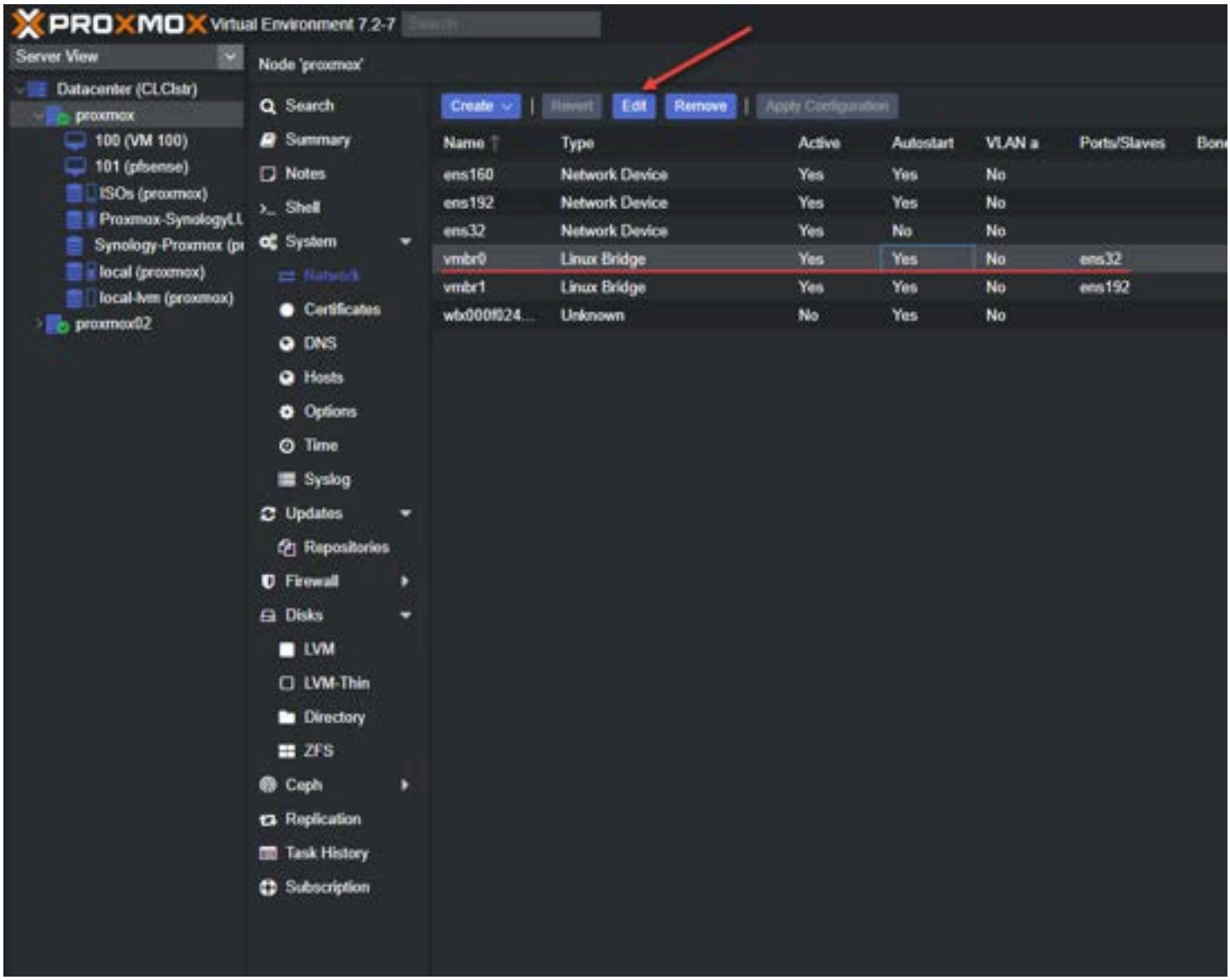
Proxmox management interface VLAN tagging

A couple of steps are involved to correctly tag VLANs from the Proxmox VE side of things. First, we must make the Linux bridge in Proxmox Server VLAN aware. This first step can be completed from the web interface.

Configuring the Linux Bridge to be VLAN aware

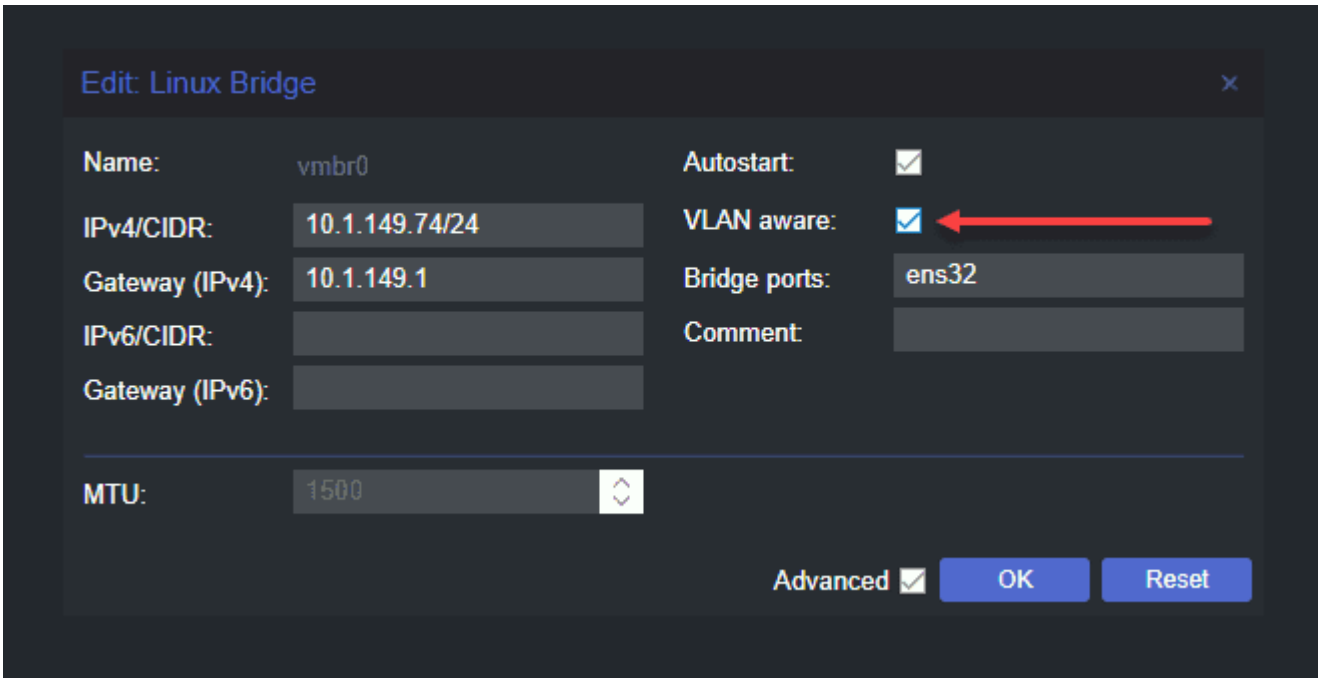
We need to navigate to the Promox host > **System > Network** and then **Edit** the properties of the default linux bridge interface in Promox.

Navigating to the default Linux bridge interface in Proxmox server and editing the default bridge in the Proxmox GUI, we click the Edit button in the user interface.



Checking the box next to VLAN aware

The first change we need to make is small. We need to tick the box next to VLAN aware. This allows us to configure Proxmox and the Linux bridge to be aware of vlan tagging for the Linux bridge interface.

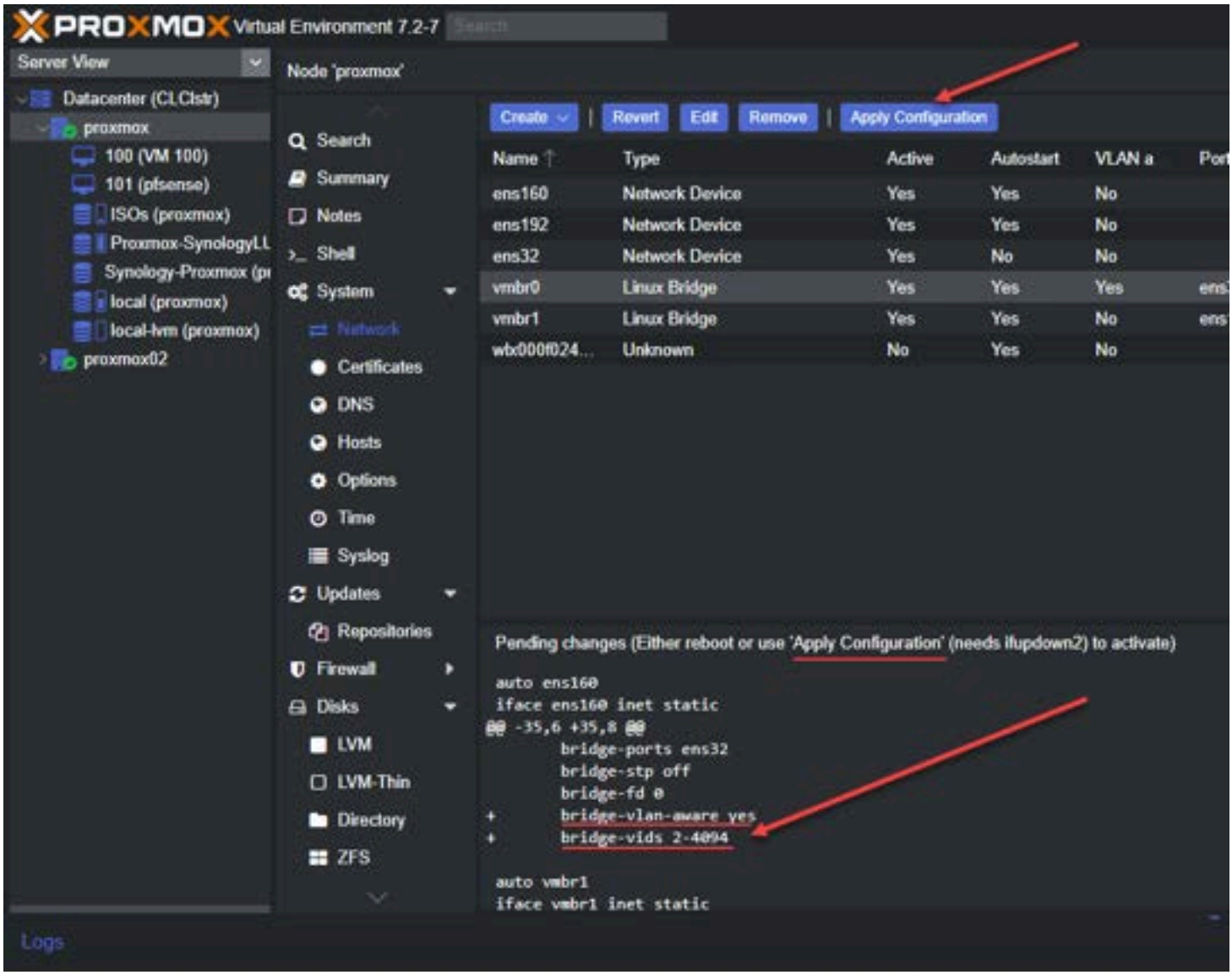


Applying the configuration

When we edit the network configuration of the Proxmox node, we need to **Apply configuration** to the network changes. This will apply the changes and restart networking services.

The Proxmox Server displays a preview of the etc network interfaces file, which shows the changes made to the default bridge interface:

```
bridge-vlan-aware yes
bridge-vids 2-4094
```



Making changes to etc network interfaces file for the new Linux bridge interface

This is the first part of the Proxmox server configuration for VLAN-aware traffic on the management VLAN for the Proxmox system. Now we need to make some low-level changes to the etc network interfaces file on the Proxmox host.

Editing the default Linux bridge

We need to edit the file to set the VLAN for the management VLAN and IP address, which is a static address to the new bridge interface tagged with a VLAN.

Below is an example of the default configuration **after** we have turned on the VLAN aware setting.

Default configuration

```
auto vmbr0

iface vmbr0 inet static

        address 10.1.149.74/24

        gateway 10.1.149.1

        bridge-ports ens32

        bridge-stp off
```

```
        bridge-fd 0

        bridge-vlan-aware yes

        bridge-vids 2-4094

auto vmbr1

iface vmbr1 inet static

        address 172.16.16.254/24

        bridge-ports ens192

        bridge-stp off

        bridge-fd 0
```

New Linux Bridge VLAN configuration

However, we want to add VLAN tagging from the management interface bridge interface. To do this, we need to change the configuration to the following. Note below, we take the IP address off the iface vmbr0 configuration or iface eno1 inet manual config. However, we leave the VLAN configuration intact. We then create another network interface that is very similar to “subinterface” configuration syntax. We create a vmbr0.<vlan tag> configuration. It is where we place the IP address configuration for the Linux bridge network device.

With this configuration, the Proxmox IP will now be the static IP address and subnet mask is configured in the new bridge interface, since these are virtual interfaces off the main Linux bridge shown with the iface vmbr0 inet manual stanza. You also place the default gateway on the new Linux bridge. You can configure multiple IP addresses across different bridges configured on your Proxmox server.

VLAN config

```
auto vmbr0

iface vmbr0 inet manual

        bridge-ports ens192

        bridge-stp off

        bridge-fd 0

        bridge-vlan-aware yes

        bridge-vids 2-4094

auto vmbr0.333

iface vmbr0.333 inet static

    address 10.3.33.16/24

    gateway 10.3.33.1
```

Network Switch network configuration

After configuring your new Linux Bridge virtual interface, we need to make sure the physical interface of the network switch port is configured as a trunk port to “understand” the VLAN tagging coming across from the Promox server. The physical port of the switch allows carrying the tagged VLAN traffic to the rest of the network.

The VLAN ID is part of the Layer 2 ethernet frame. If the physical interface of the switch port is not configured correctly, VLAN traffic for the VLAN ID is discarded.

Virtual machines VLAN traffic

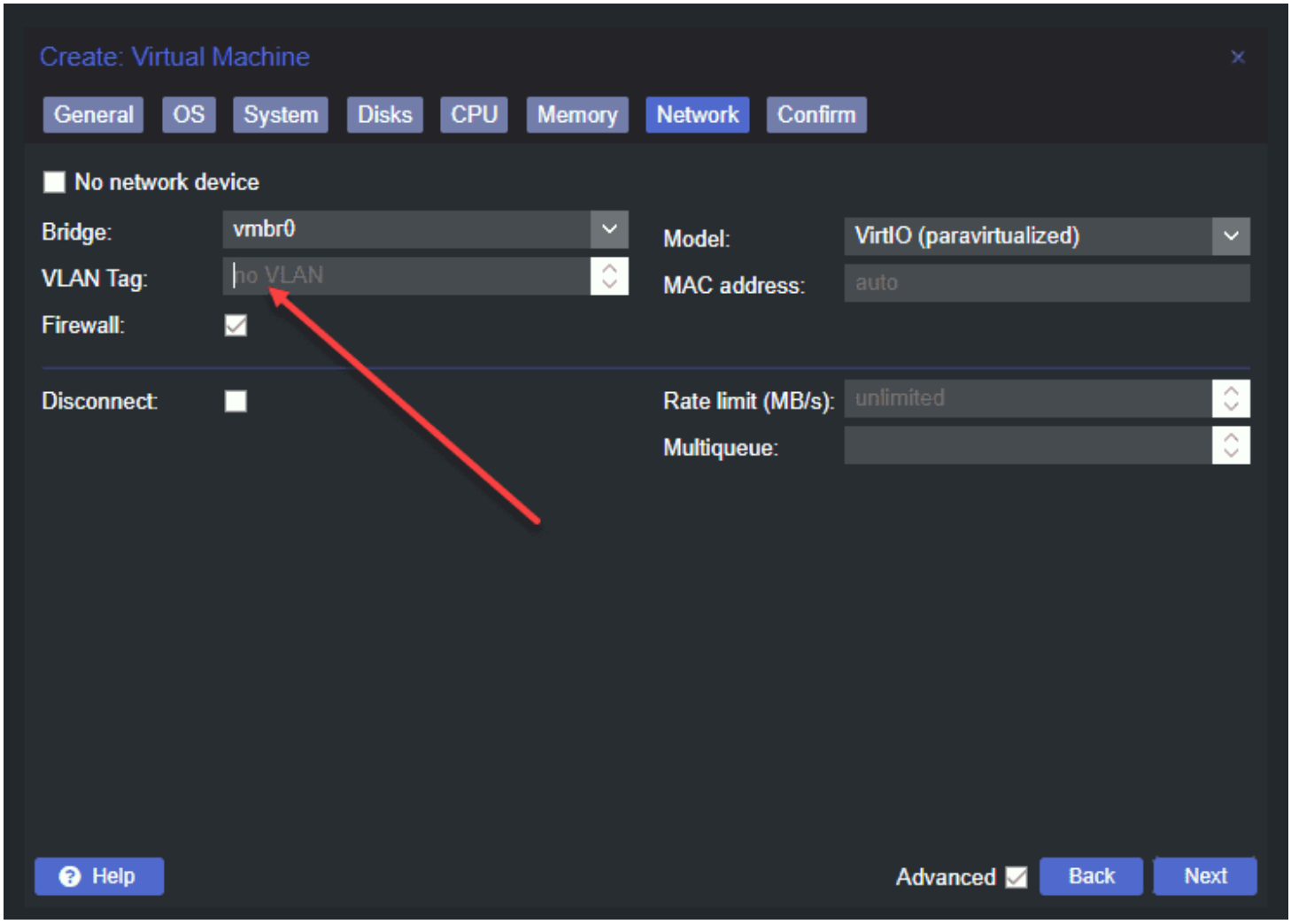
Once we have made the default Linux bridge VLAN aware, virtual machines can also have a VLAN tag associated with their network configuration. It allows the virtual machine to tag VLAN traffic and be placed on that particular VLAN.

Creating VM with the VLAN tag

When you create VMs, you can choose to tag the network traffic with a VLAN ID. This allows sending the virtual machine traffic through the physical device VLAN interface to the rest of the physical network.

The beauty of the VLAN aware bridge is you can have other VLANs configured on other virtual machines, and each can communicate on the required VLAN interface.

Below is an example of the screen to create a new VM and the networking screen. The VLAN tag field allows typing in your VLAN interface number.



Home Lab network configuration

Proxmox is a great choice for building a home lab environment and run VMs on your home network. Once you create your new VLAN in Proxmox server and on your network device, you can start building out your lab environment and have traffic flow as expected. You can make sure your virtual machines are connected to the appropriate networks.

You can also ensure you have Internet access via inter-VLAN routing on your network switch, firewall, router, etc. VLANs create a lot of flexibility from a physical cabling, ports, and virtual configuration, providing many opportunities to allow traffic to flow from your VM guests or physical hosts.

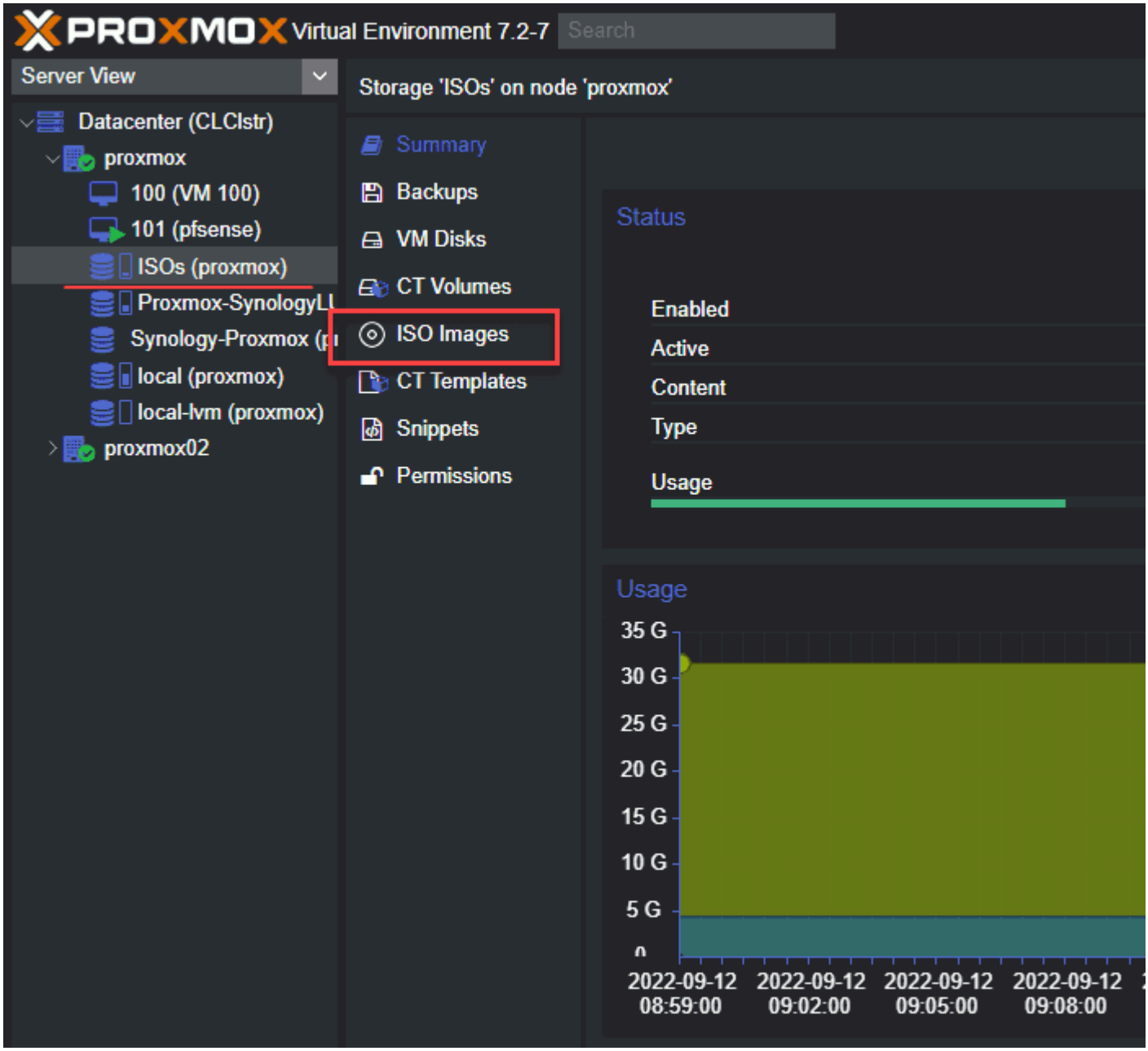
Proxmox resources

Take a look at my Proxmox resources that I have written about below:

- Proxmox vs ESXi – ultimate comparison 2022
- pfSense Proxmox Install Process and Configuration
- Proxmox Update No Subscription Repository Configuration
- Proxmox iSCSI target to Synology NAS
- Nested Proxmox VMware installation in ESXi

Proxmox Create ISO Storage Location – disk space error

September 12, 2022
Proxmox



6120339f af78 44c4 8394 7899a5d5d08a

If you are working with Proxmox in your home lab or otherwise, one of the first things you will want to do is upload ISO installation media to your Proxmox host. You can mount a physical CD to your Proxmox host, of course. However, this is cumbersome and not feasible for remote configurations and installing a wide range of operating systems across the board in the Proxmox environment.

Uploading ISO installation media to your Proxmox host is the way forward for most. If you are like me, you may run into issues with a default installation of Proxmox and the partition size configured for ISO images by default. Let's talk about Proxmox create ISO storage location and see how this is completed.

Take a look at the Youtube video walkthrough of the process below:

Proxmox Create ISO Storage Custom Location - Solve disk space error
https://youtube.com/watch?v=ZsVJkDoK-hg

Proxmox VE Server and installing operating system guests

Promox VE Server is a great open-source hypervisor that provides many capabilities and features. It has the capability as a native feature to upload ISO files. You can then select file to select the ISO image you want to use to install guest operating system virtual machine instances.

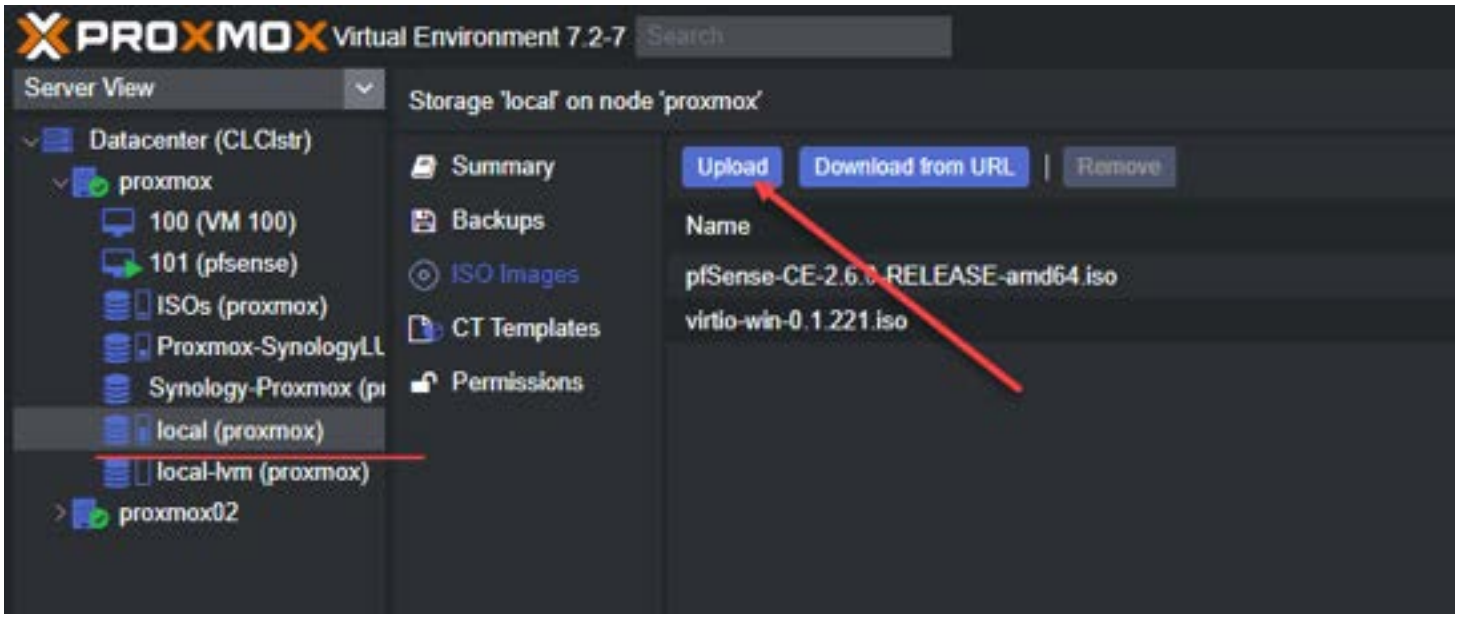
An ISO file is a disk image that most software vendors provide to install operating systems. This includes Linux operating systems like Ubuntu and also Microsoft Windows operating system variants.

When you configure the virtual machine in Proxmox VE, you select the disc image and the ISO image is used as part of the virtual machine installation process.

Uploading Proxmox VE ISO images to the server

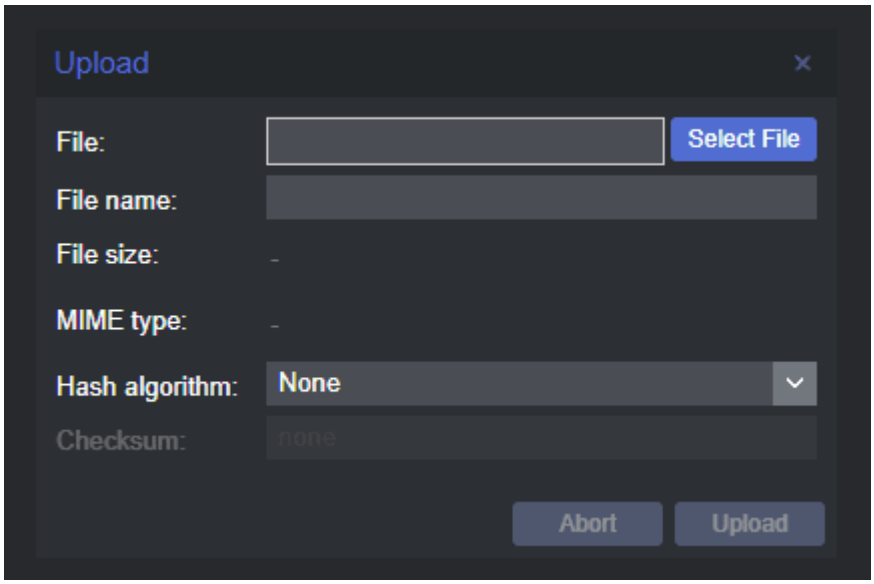
Like VMware and other hypervisor solutions, the Proxmox VE server has the means to upload ISO files using the Server view interface.

In the Server View, click the storage pool location **> ISO images > Upload**.



Uploading an ISO to Promox VE server

It will launch the box to upload the ISO files. Browse to your ISO file and click the **Select file** button to point to the ISO image you want to use and click finish.

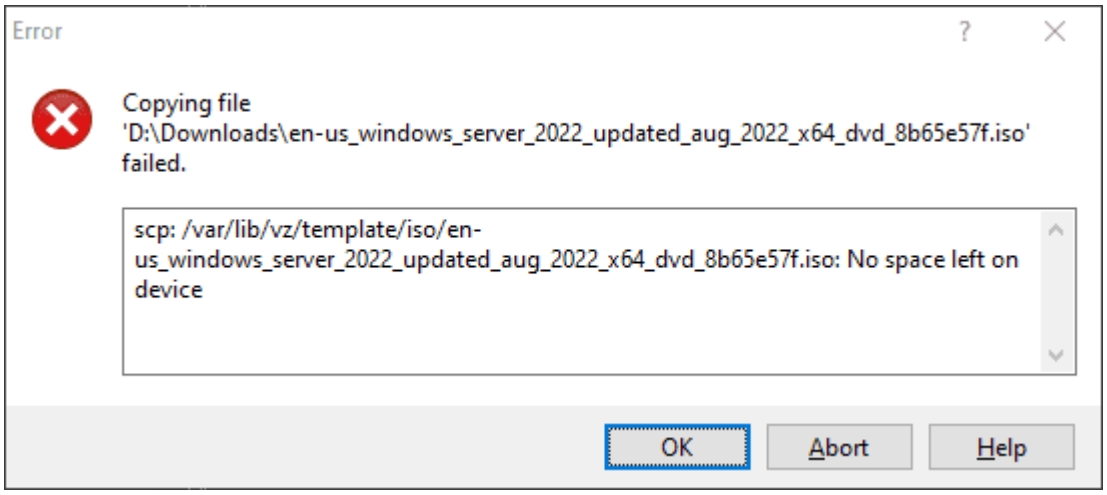


Select the ISO file and then Upload

As you can see, the upload can be perfromed from the browser, like other hypervisors, such as VMware.

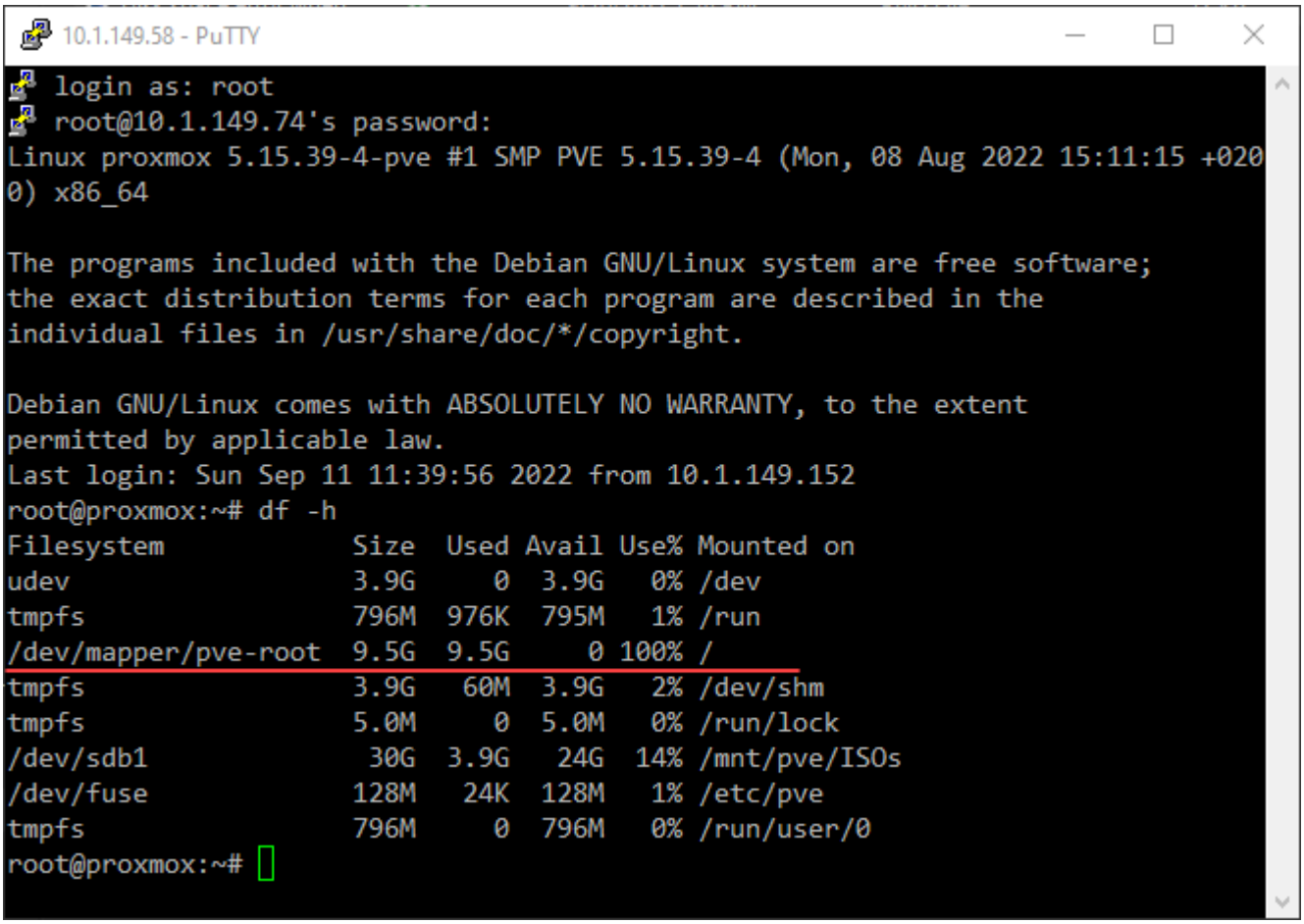
Proxmox VE ISO files storage disk space error

However, the following disk space error may be one issue with a default Proxmox VE installation.



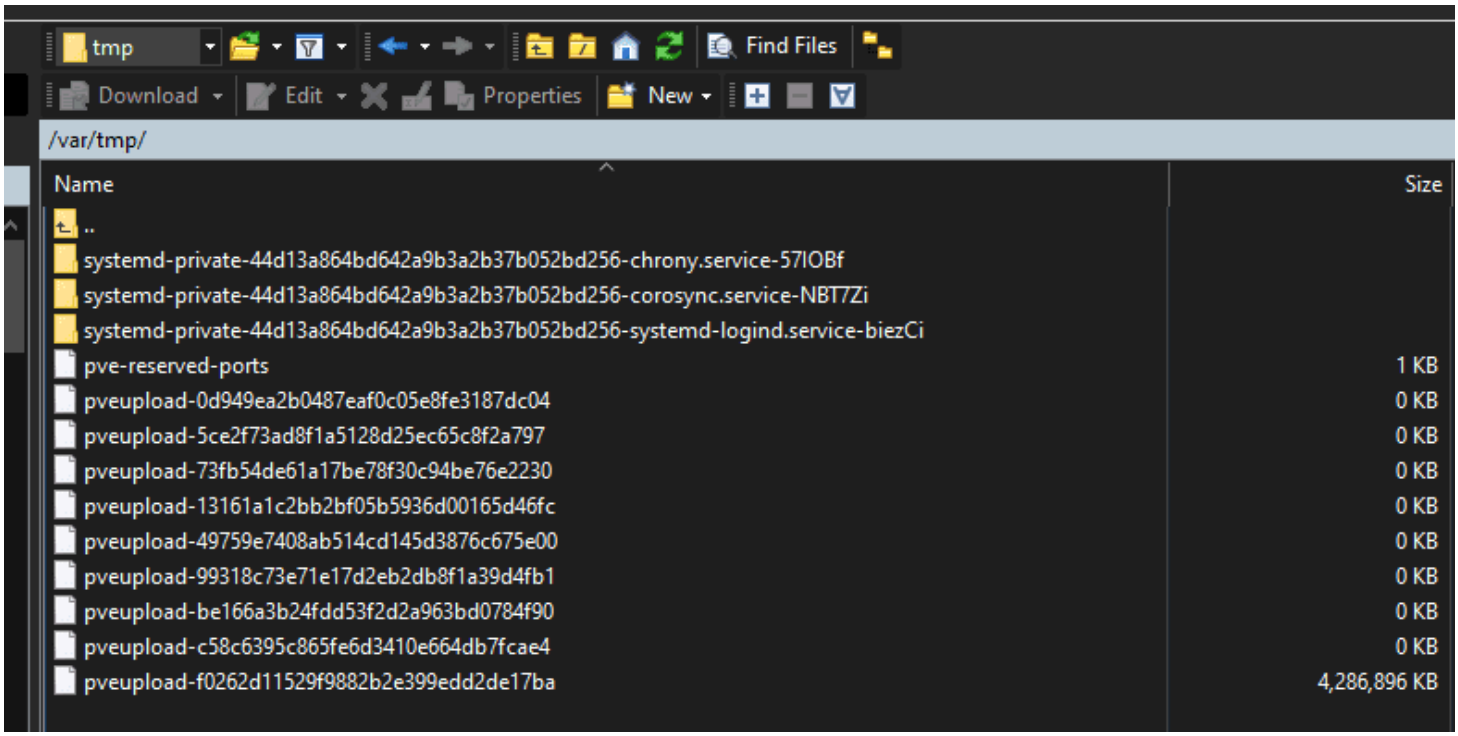
Disk space error when uploading ISO images to Proxmox VE

The default Proxmox VE installation storage pool space only included a 10 GB partition for uploading ISO files. When you upload an ISO image to a Proxmox VE server, it will first attempt to upload the ISO image file to the **/var/**



Viewing disk space on a Proxmox VE host

As you can see below, the image file is first uploaded to **/var/tmp** before they are staged into the permanent location found at **/var/lib/vz/template/iso** folder.



Uploaded image files in the tmp directory in Proxmox VE

Proxmox Create ISO image Storage Location

How do we create a custom location for ISO storage location for storing your ISO image files in Proxmox VE?

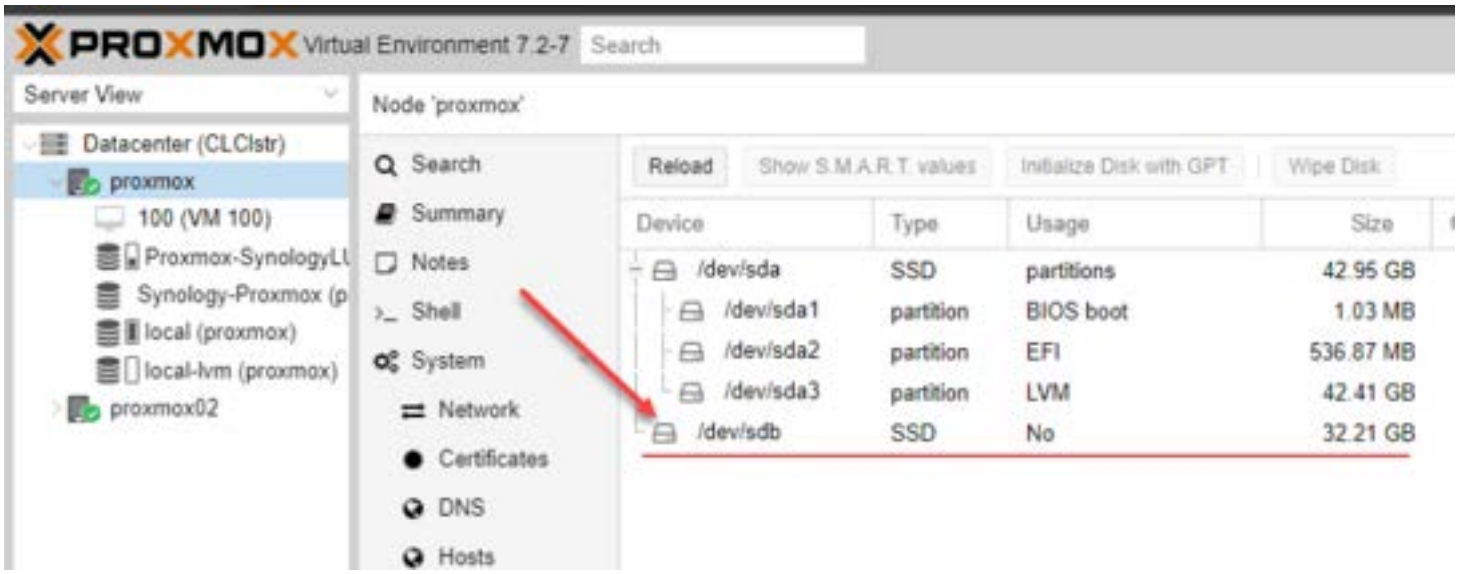
First of all, we need to go through a directory creation process to create a custom location for uploading your operating system ISO files to your Proxmox VE server for creating your Server VM installation or other operating system VM installations.

The process involves the following steps:

1. Add a new hard disk to your Proxmox host
2. Create a new Proxmox directory
3. View the storage location in your Proxmox server web interface

Add a new hard disk to your Proxmox host

The first step in adding an ISO image storage location in Proxmox is to add an additional hard disk where we can create a directory. Below, I have added a new hard disk to the Proxmox host. We can use this in the configuration of the new ISO image storage location, instead of the /var/lib/vz location on the screen.



Adding a new hard disk to Proxmox VE server host

Viewing the new disk from the command line

You can view the new disk from the Promox command line using the command and switch:

df -h

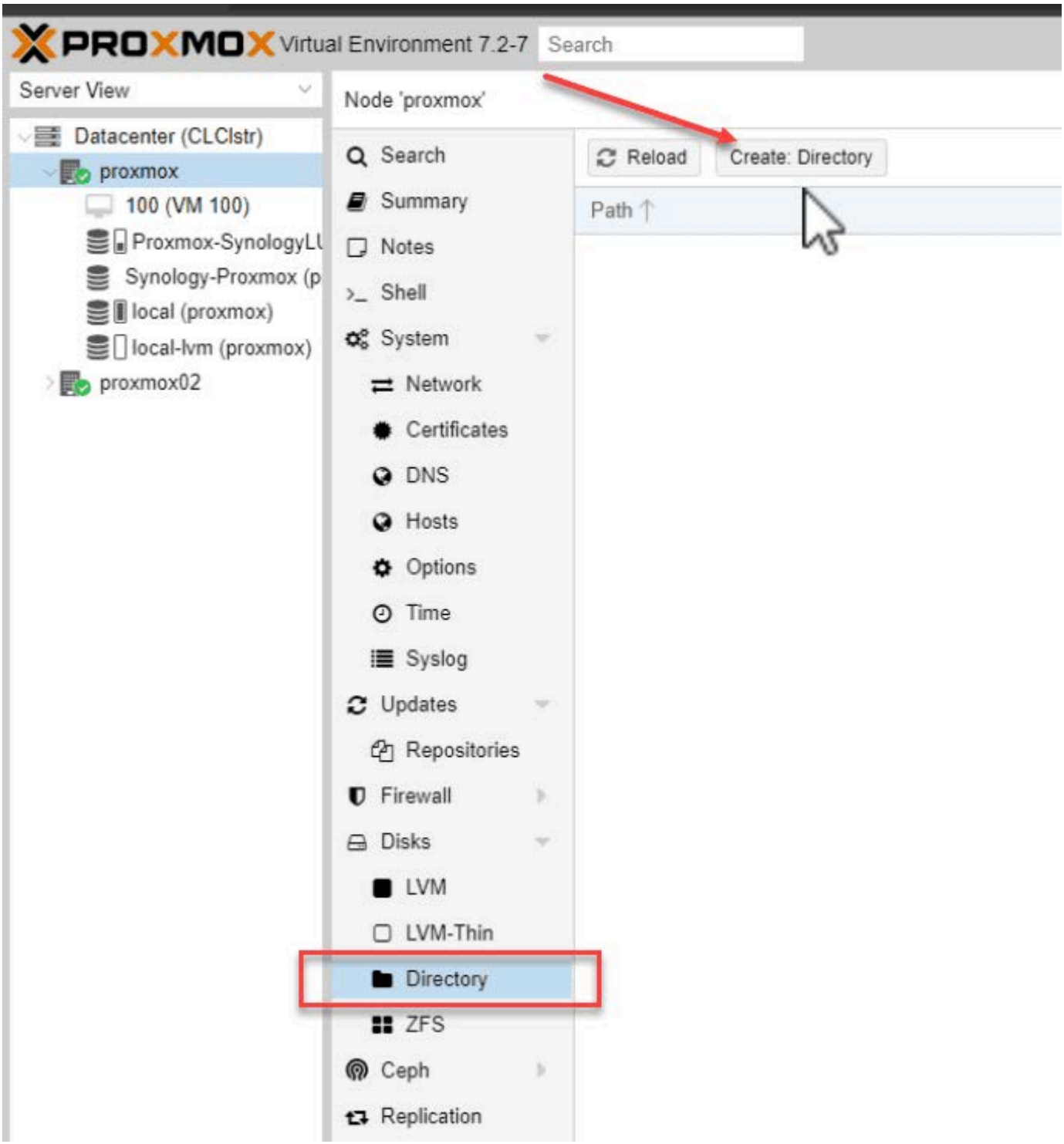
This is the same command you would use in Ubuntu or other Linux distribution for viewing disk space for the store.

```
root@proxmox:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
udev                  3.9G     0  3.9G   0% /dev
tmpfs                 796M  984K  795M   1% /run
/dev/mapper/pve-root  9.5G  5.4G  3.7G  60% /
tmpfs                 3.9G   66M  3.9G   2% /dev/shm
tmpfs                 5.0M     0  5.0M   0% /run/lock
/dev/sdb1              30G  4.2G   24G  15% /mnt/pve/ISOs
/dev/fuse             128M   24K  128M   1% /etc/pve
tmpfs                 796M     0  796M   0% /run/user/0
root@proxmox:~#
```

Viewing the local storage on a Proxmox VE server

Create a new Proxmox Directory for ISO image upload iso files

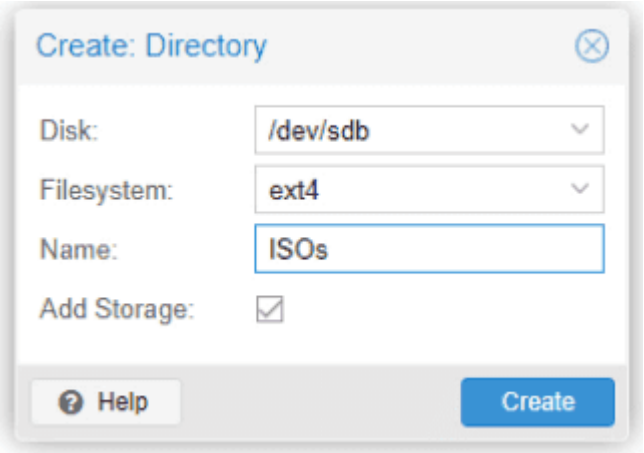
Next, we navigate to the **Disks > Directory > Create Directory** button in Proxmox. Here we can create the directory we need and format the file storage for uploading ISO files.



Create a new directory in Proxmox for ISO storage

When you click the **Create directory** button, you will see the following Create Directory dialog box. Select the disk, filesystem, name, and check the box to **Add storage**. Then click the Create button.

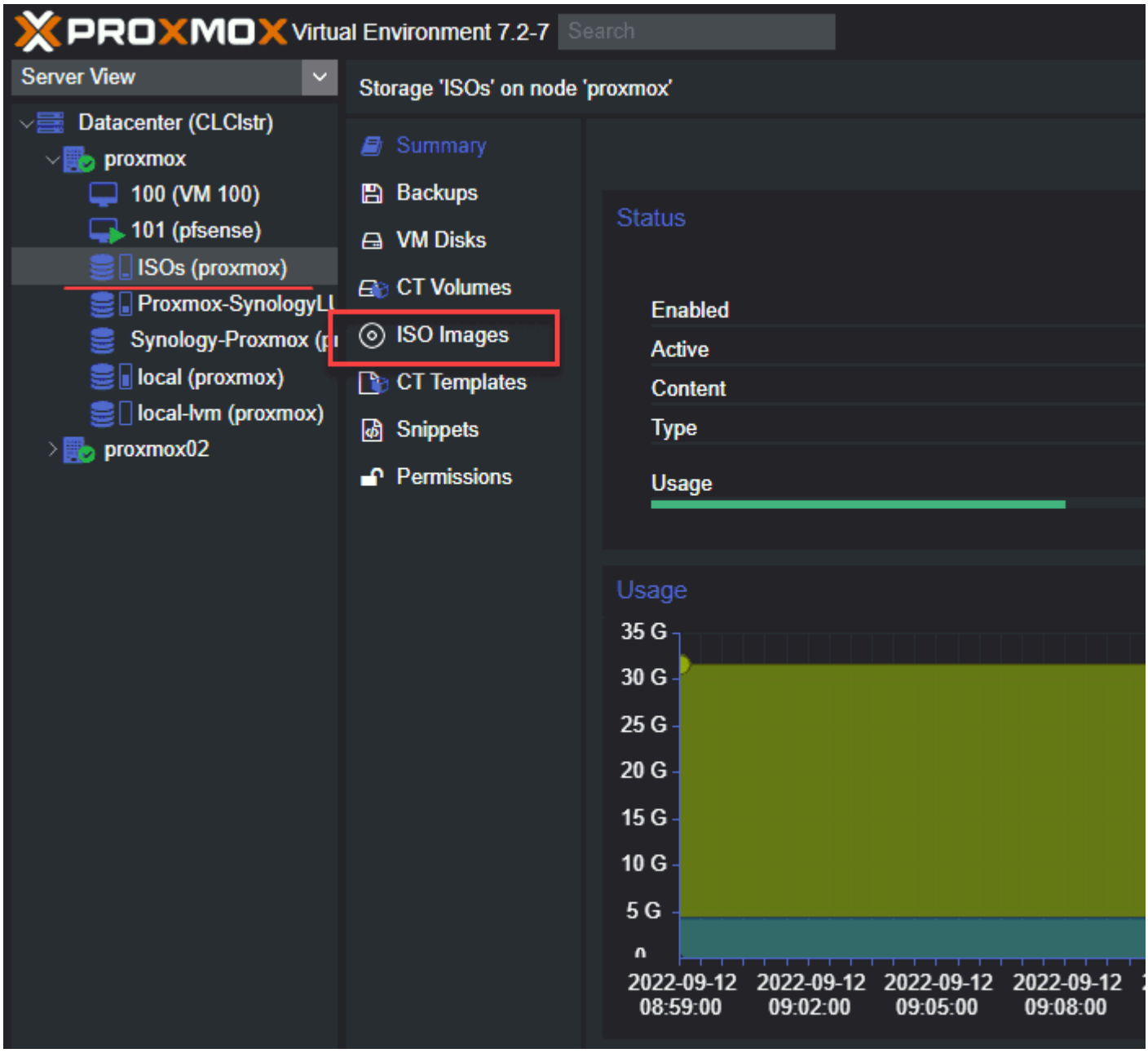
Below is an example of creating a new ISO image storage location on a Proxmox server host. When you create the new directory, you can then ensure your ISO images are stored in the new location when uploading ISOs for creating your new VM hosted in Proxmox.



Create the directory for Proxmox ISO image files

Viewing the new Proxmox ISO image location

After creating the new storage location, you will see the ISO image location listed in your Proxmox browser interface on the left-hand side.



New Proxmox ISO storage location for storing ISO image files

Proxmox storage FAQs

What is Proxmox? Proxmox is an open-source hypervisor that allows easily running virtual machines and containers.

What is Proxmox ISO storage? This is storage in Proxmox allowing you to upload ISO files to storage and use these to install VM guests in Proxmox.

How do you upload ISO files to Proxmox server? You can do this using a web browser logged into your Proxmox host, or you can use SCP and an SCP utility like WinSCP.

Why might you get a disk upload error? If you use the default ISO storage location, you may receive the error on screen when uploading a large iso file operating system installations such as Windows Server 2022.

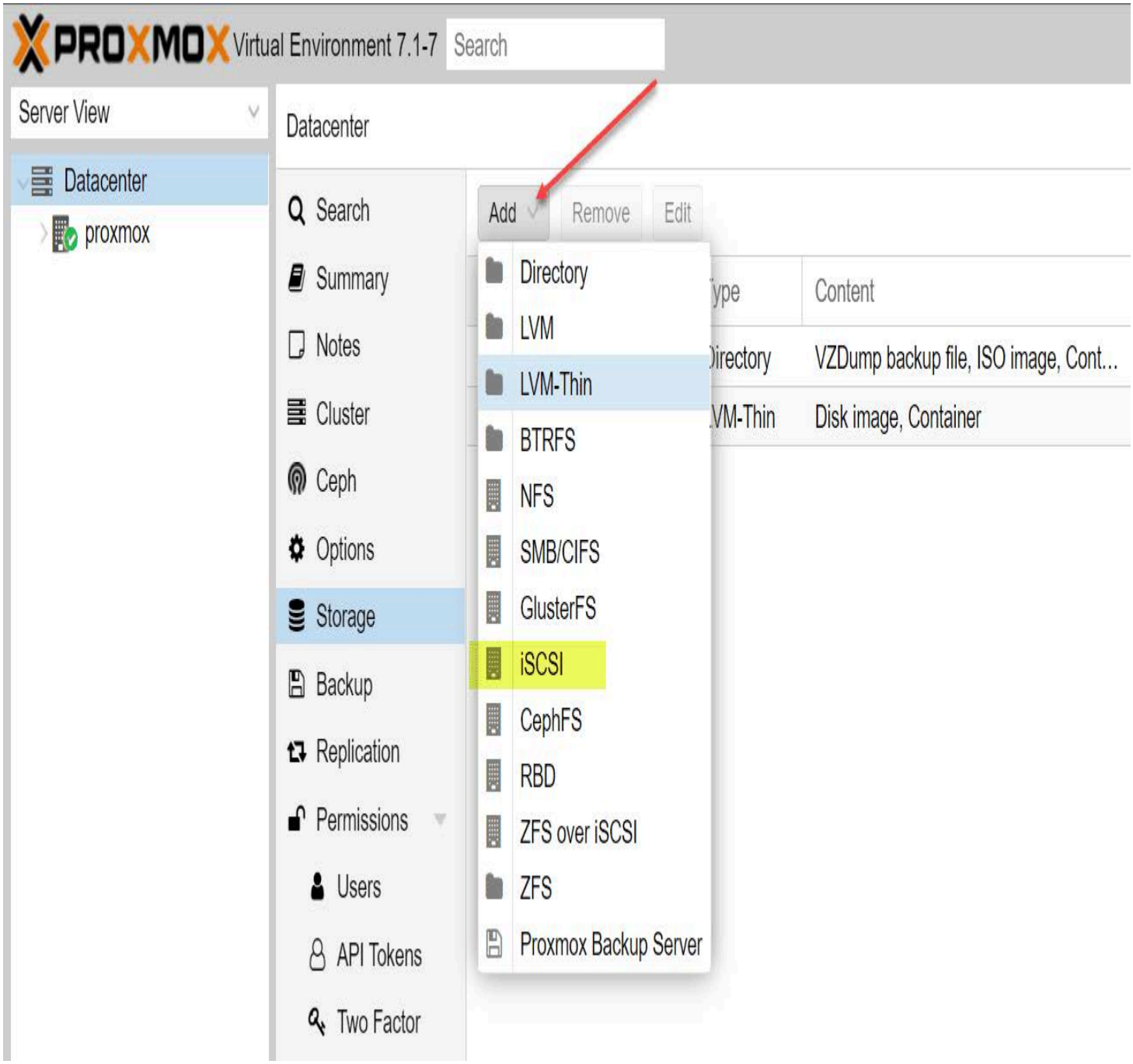
Wrapping Up

When learning about Proxmox, uploading ISO files to your Proxmox VE server is one of the first steps you will take when loading operating systems on your Promox host. If you want to learn about installing Proxmox as a virtual machine in VMware, you can look at my previous article covering that topic here.

Be sure to comment if you have alternative ways of handling the uploading and creation of ISO image file storage in Proxmox.

Proxmox iSCSI target to Synology NAS

January 19, 2022
Proxmox



Add a new iSCSI target in Proxmox

Not long ago, I wrote a quick blog post detailing how to install Proxmox inside a VMware virtual machine. However, to really play around with the hypervisor, it is great to have storage to work with. I could've added a local disk to the VM. However, iSCSI sounded way more interesting, especially with the new addition of the Synology DS1621xs+ in the home lab environment. Let's take a look at adding Proxmox iSCSI target to Synology NAS LUN and see what this process looks like.

Take a look at the video walkthrough of this process here:

Proxmox iSCSI target with Synology NAS shared storage and troubleshooting
https://youtube.com/watch?v=g5fhCiAETSU

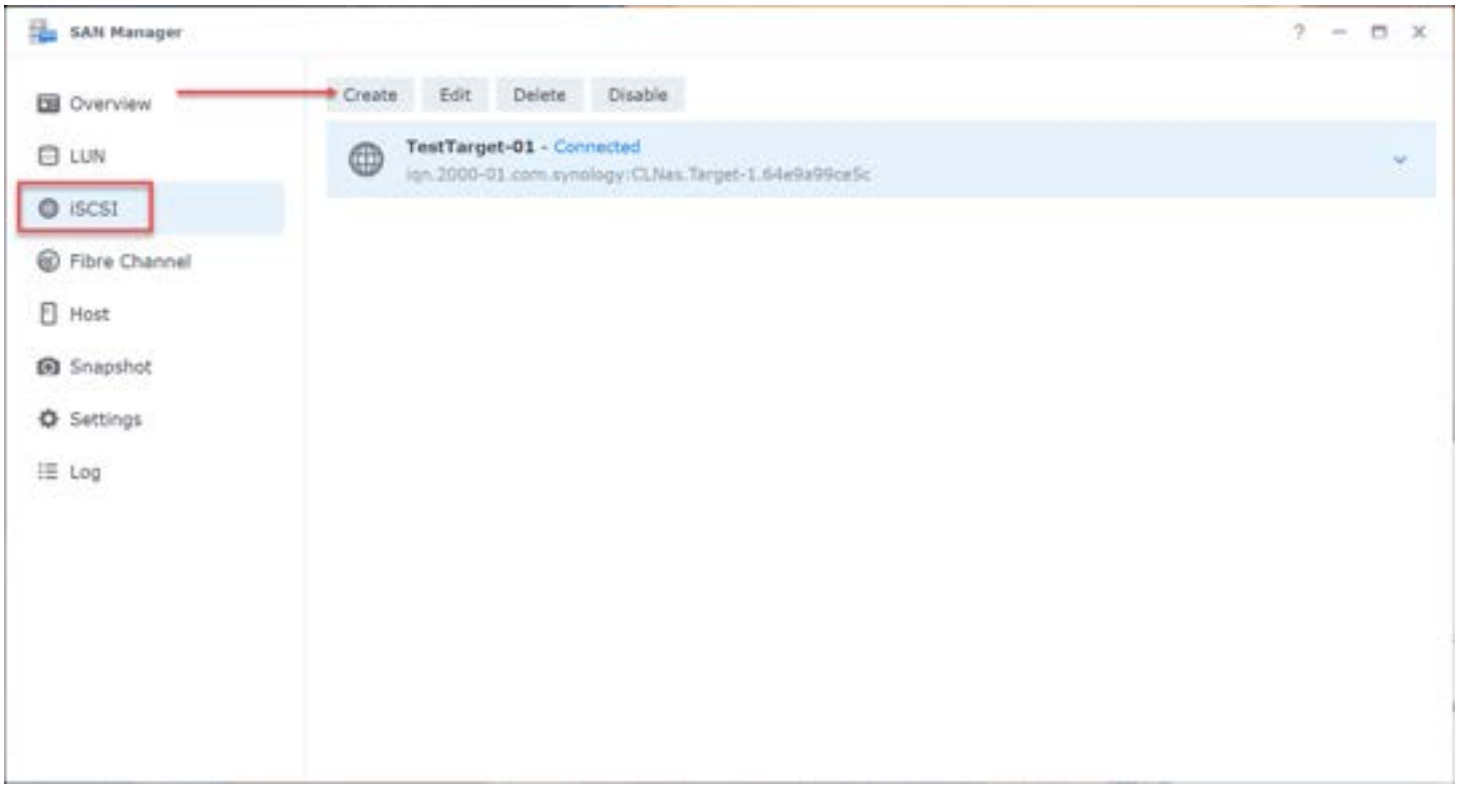
Proxmox iSCSI target to Synology NAS

The steps to complete adding a Synology on Proxmox hypervisor server looks like the following:

1. Create the iSCSI target on the Synology NAS
2. Add a dedicated interface to your Proxmox server (if you don't have already)
3. Add the iSCSI target to Proxmox
4. Create the LVM to the Synology iSCSI target

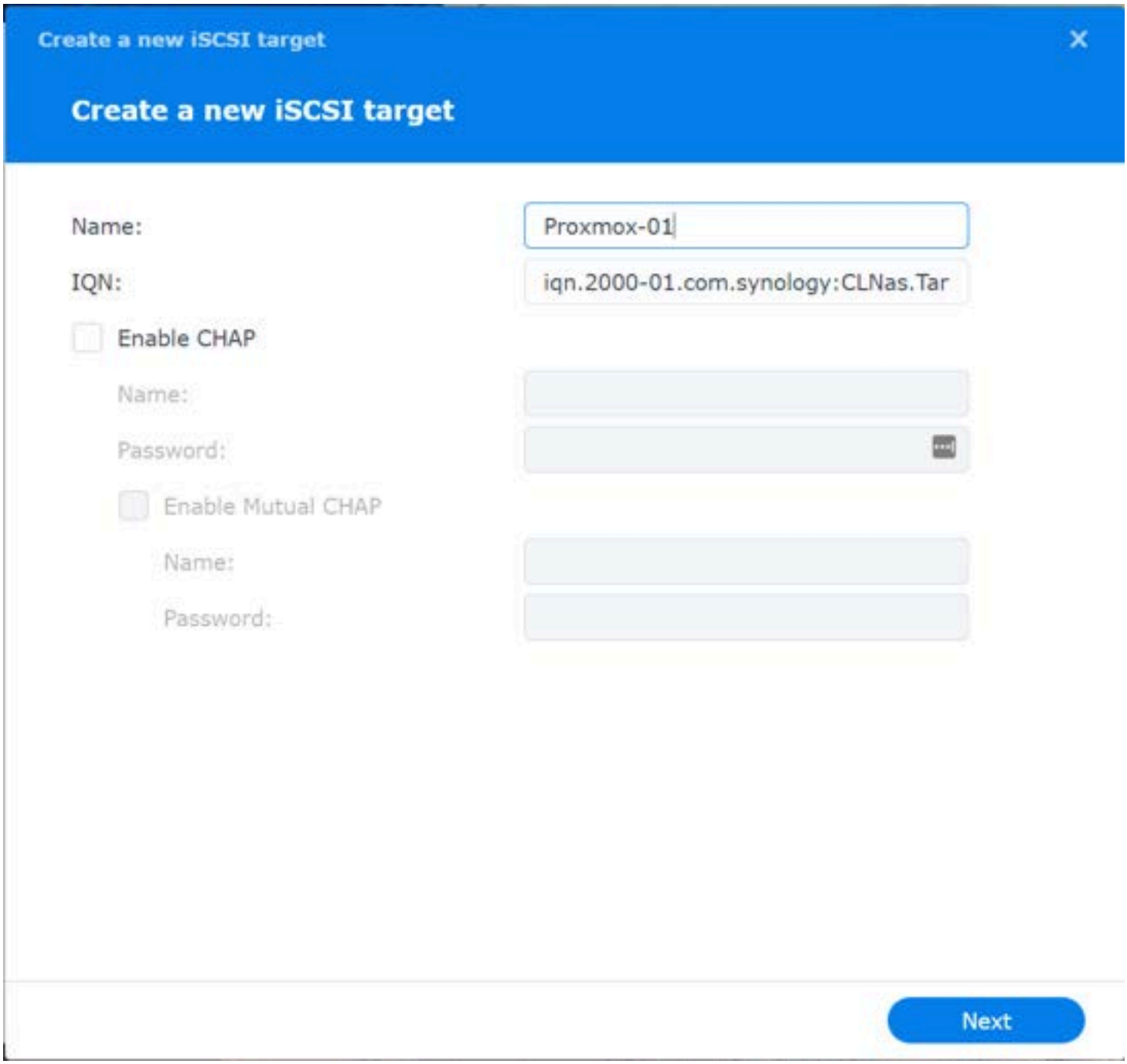
1. Create the iSCSI target on the Synology NAS

Let's first create the iSCSI target on the Synology NAS device. This process is carried out in the Synology SAN Manager. Launch SAN Manager and click **iSCSI > Create**.



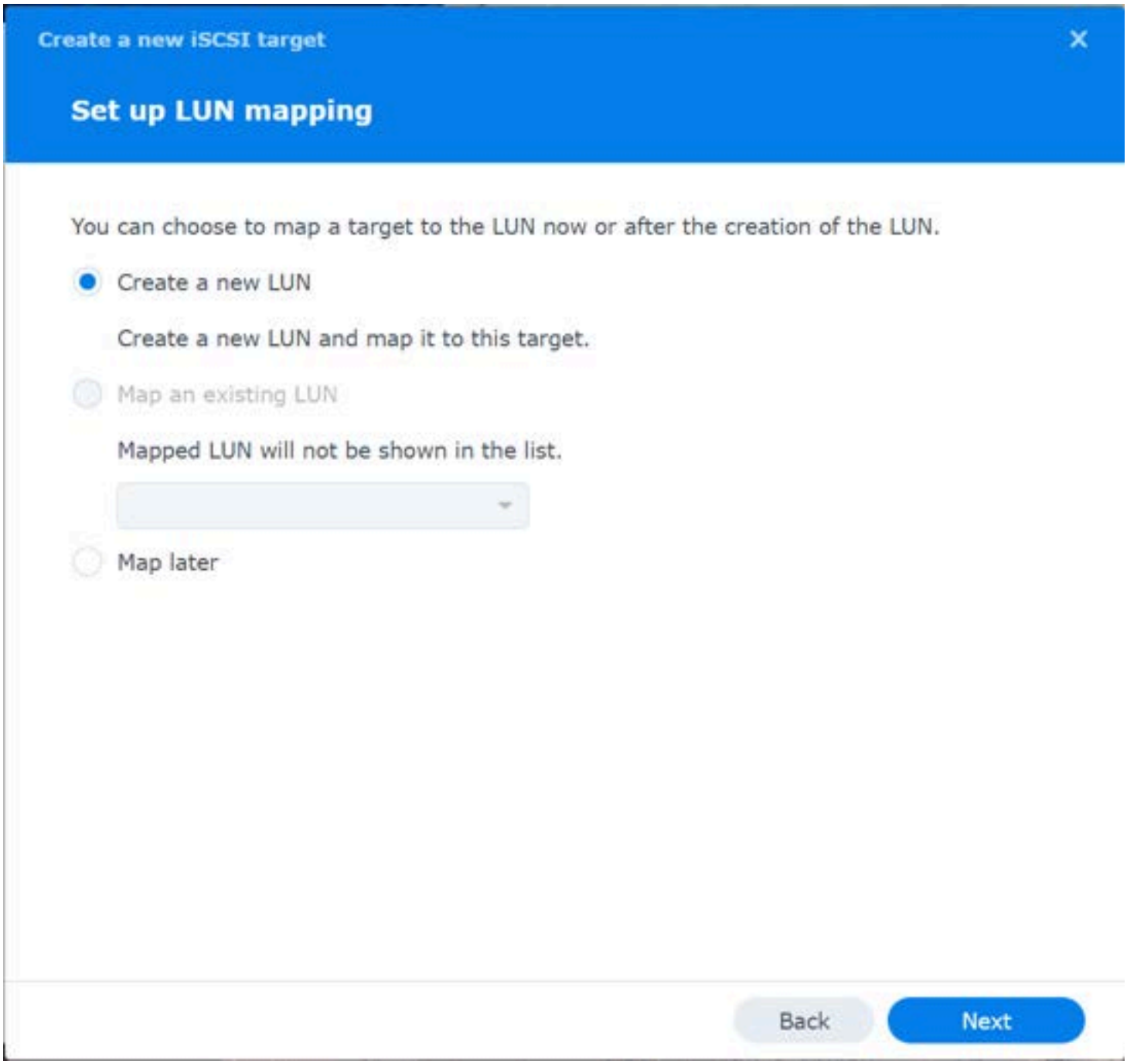
Create a new iSCSI target in the Synology SAN Manager

Configure a name for the iSCSI target and configure CHAP if you are using CHAP to secure the connections. For this test, I am leaving CHAP unchecked.



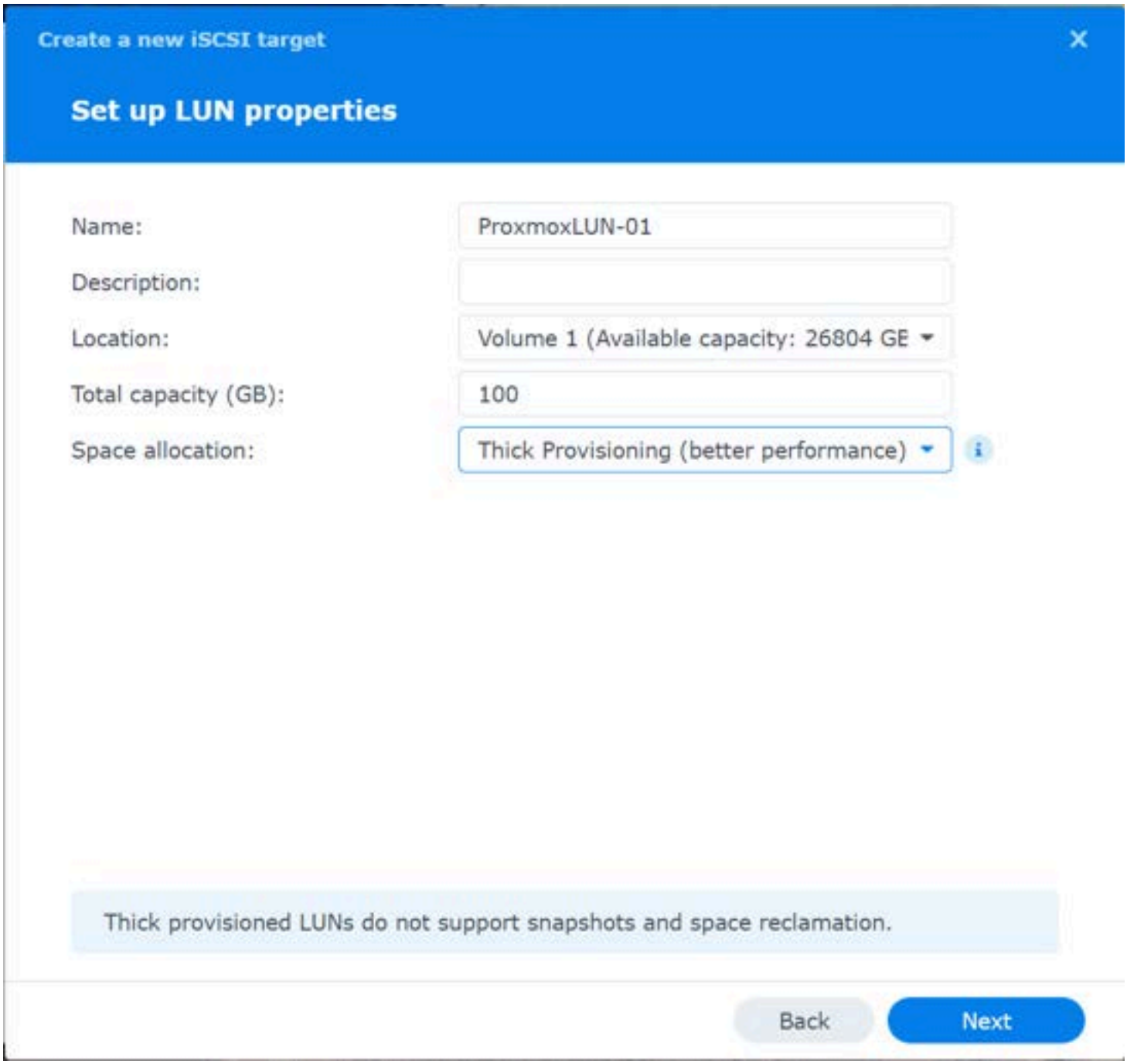
Name the new iSCSI target and choose CHAP options

From the new iSCSI target wizard, it will prompt you to create or map to a LUN. I am creating a new LUN here.



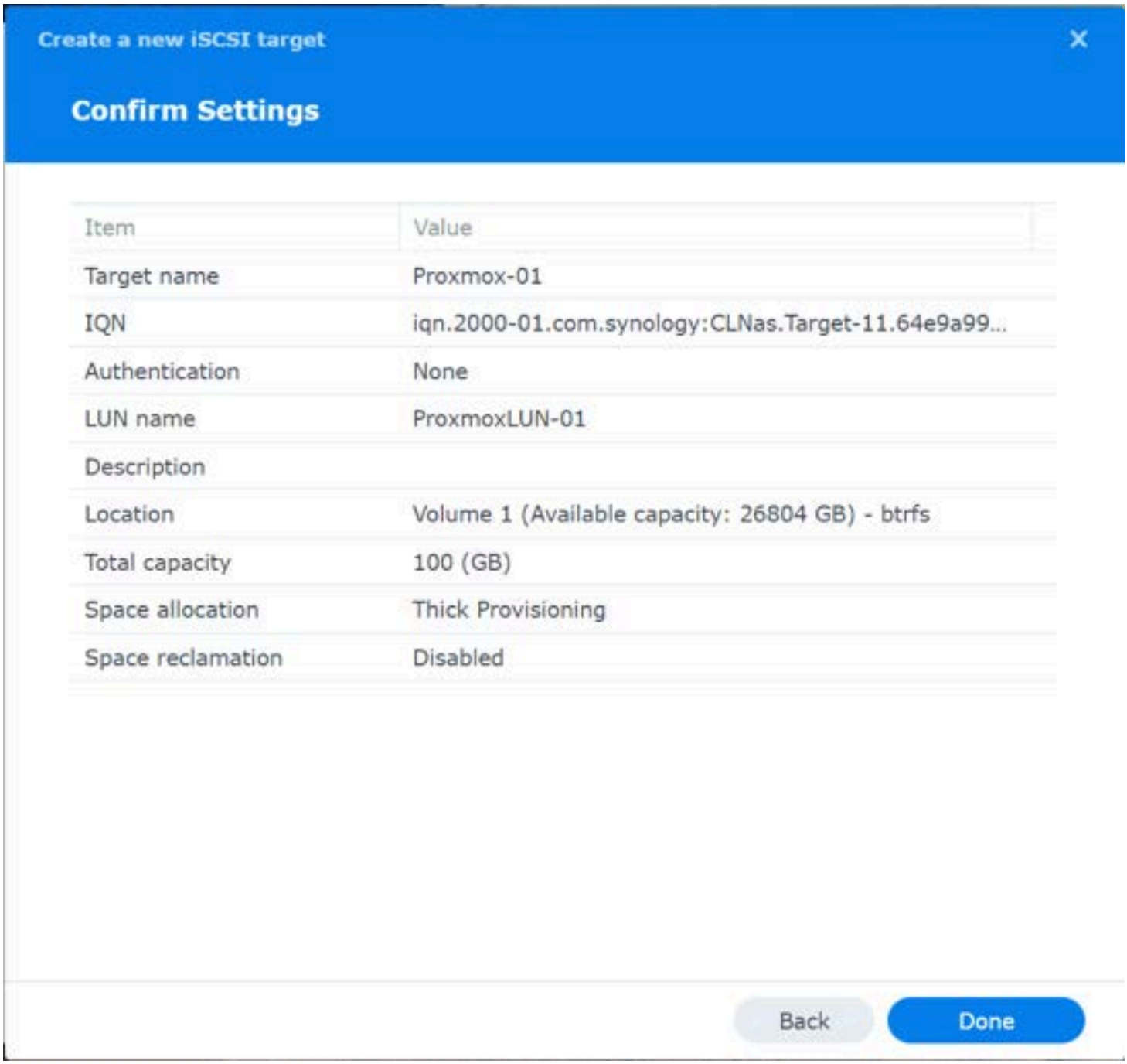
Set up LUN mapping in Synology SAN Manager

Name the new LUN and configure the **Capacity** and the **Space allocation** method (thick or thin).



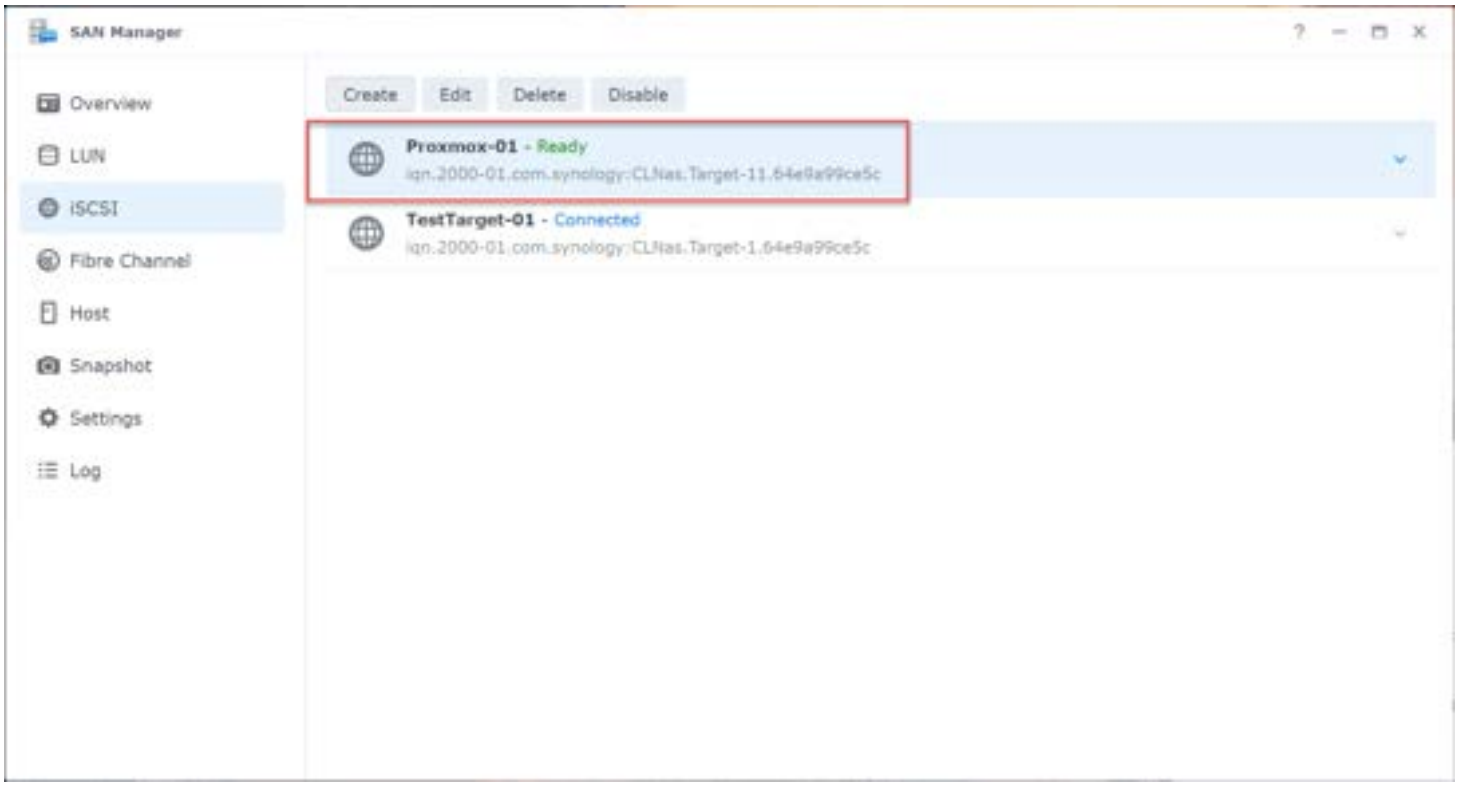
Set up LUN properties for the new LUN

Review the settings configured and click **Done**.



Confirm the settings for the new iSCSI target and LUN properties

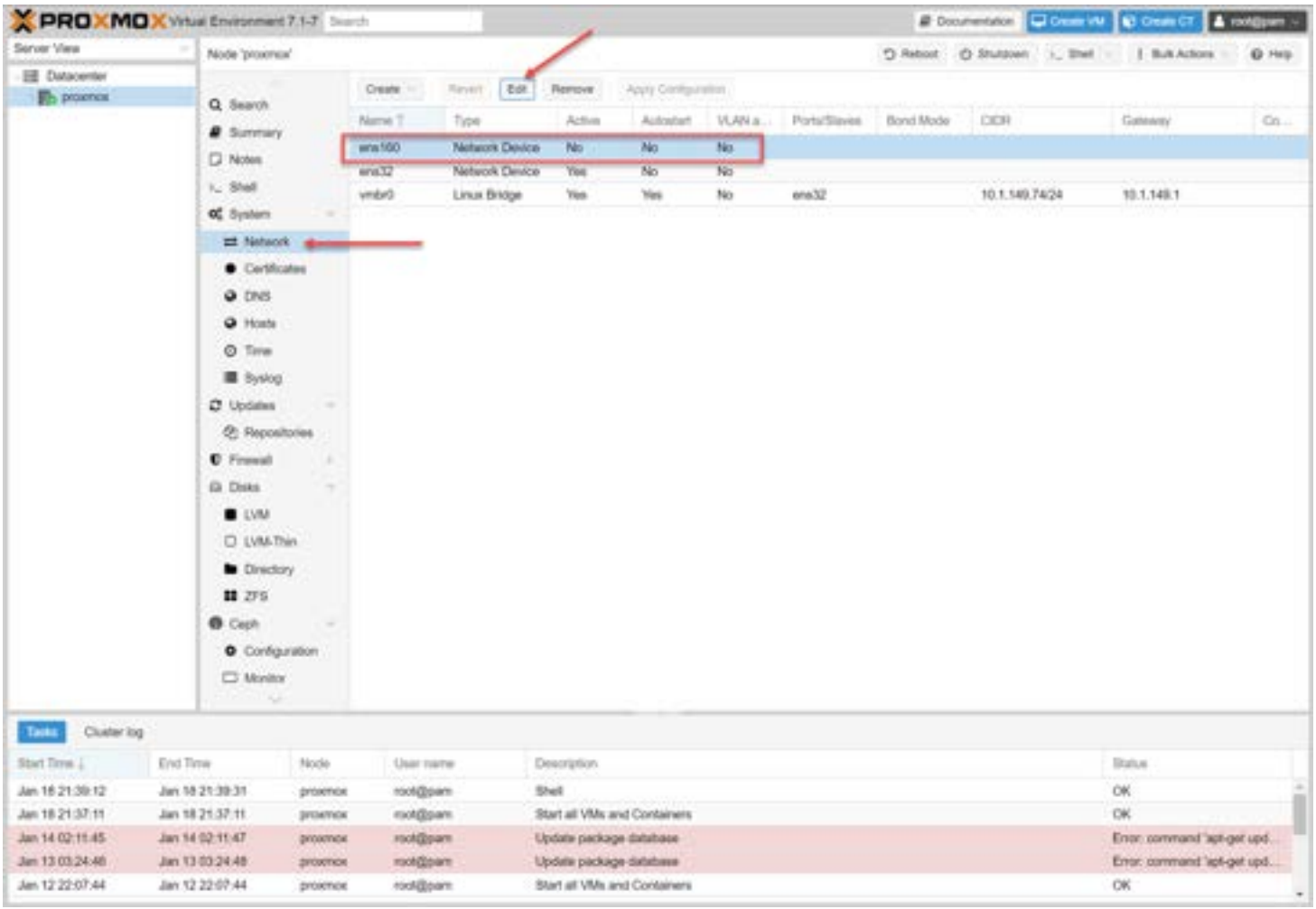
You will see your new LUN displayed in the list.



The new LUN is displayed in SAN Manager

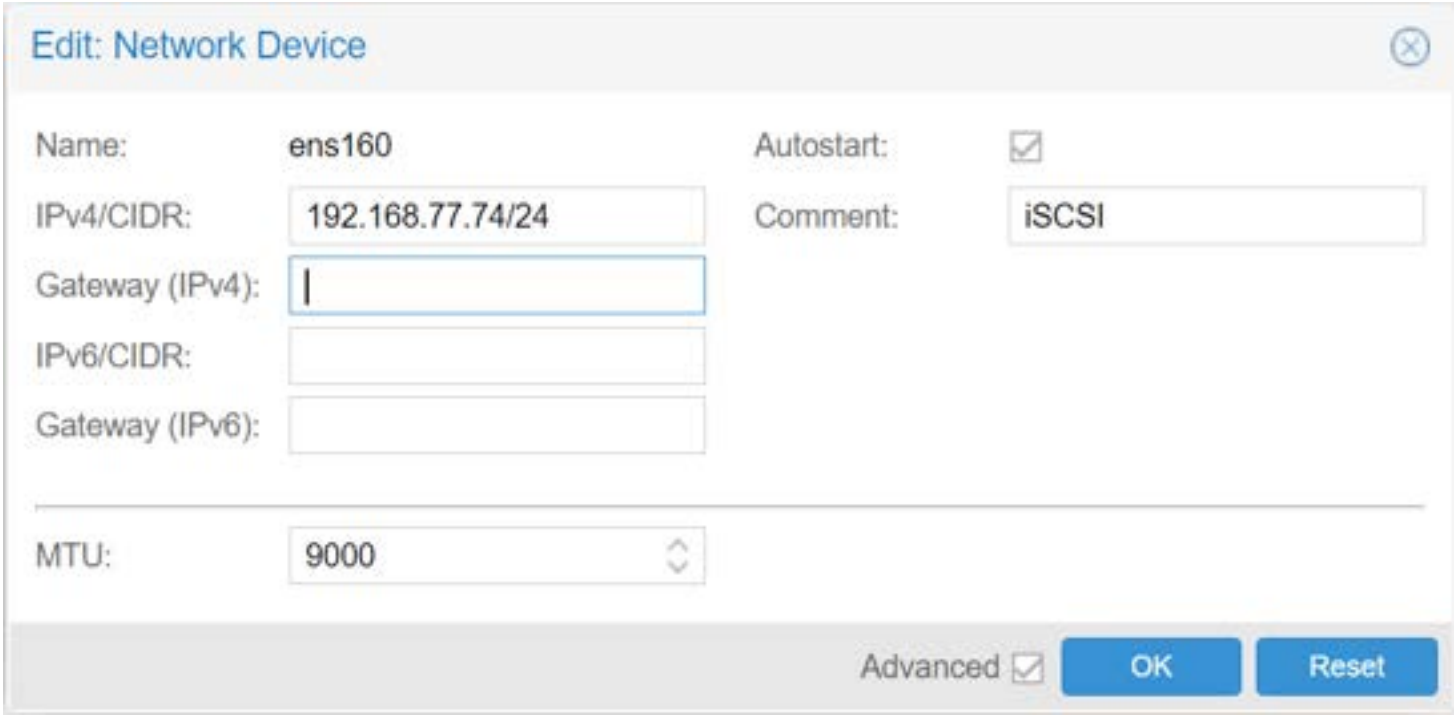
3. Add a dedicated interface to your Proxmox server (if you don’t have already)

On the Proxmox virtual machine, I have added a secondary NIC to the VM for dedicated iSCSI traffic. Now, we need to configure the NIC with an IP address. To do this, in the Proxmox GUI, click your host **> Network > <your network adapter> > Edit**.



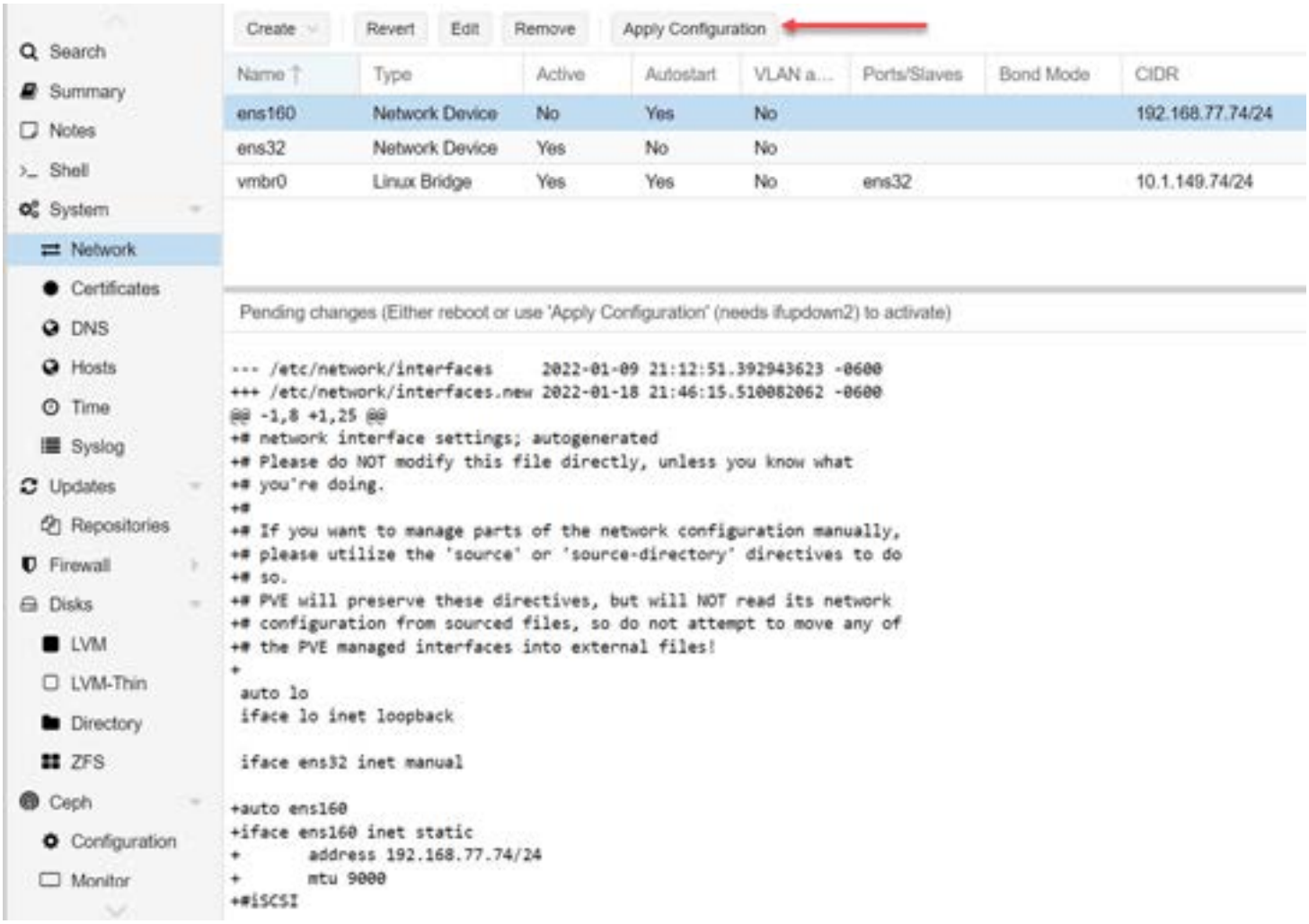
Editing the network adapter properties in Proxmox GUI

Enter the IP address you want to configure to communicate with your iSCSI target on the Synology NAS.



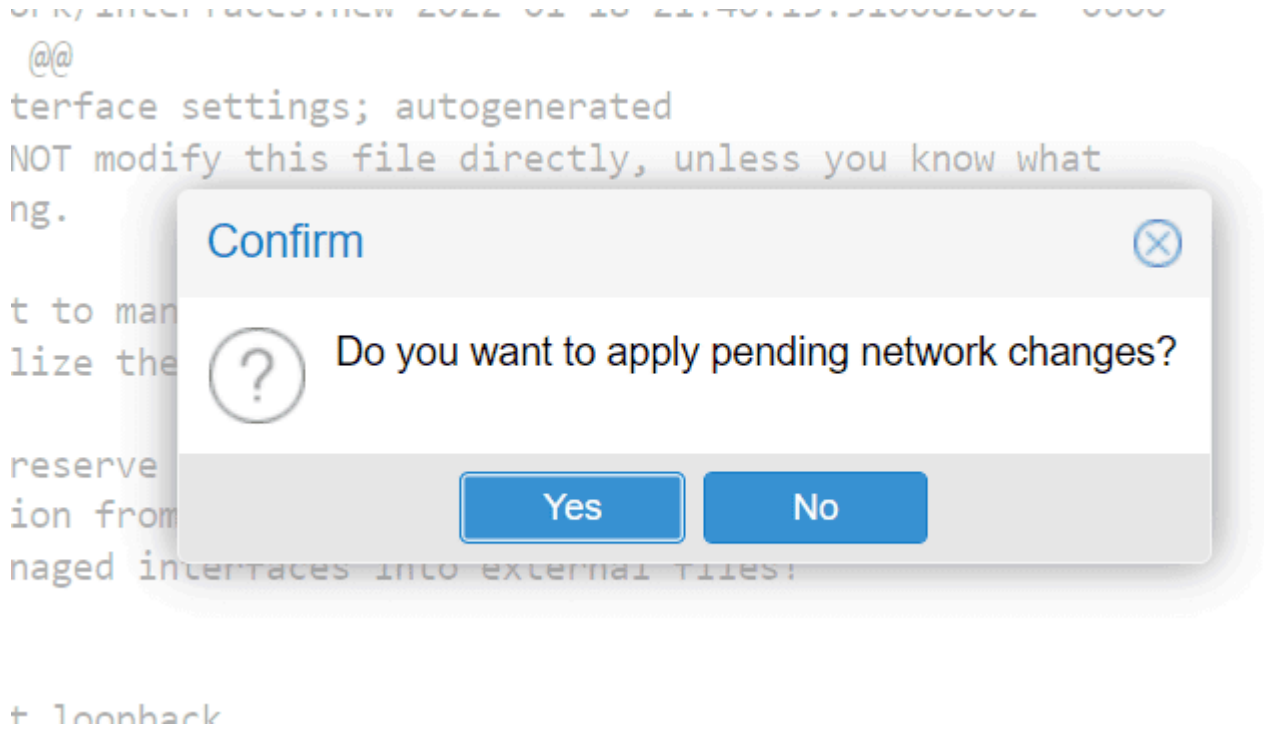
Configuring the IP address mask and comment

The IP address has been configured. However, the configuration needs to be applied as the Adapter Active status shows **No**. Click the **Apply Configuration** button at the top.



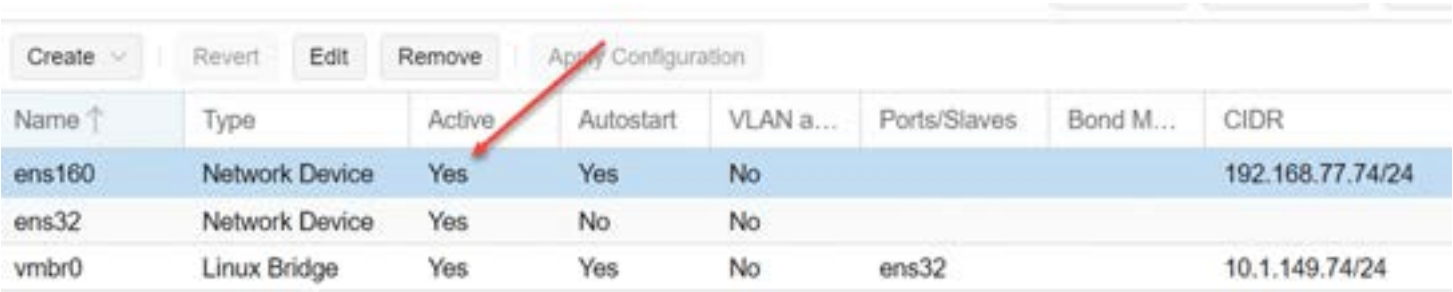
IP address is successfully configured

Confirm the new network settings.



Confirm to apply the pending network changes

The new network adapter with the configured IP address now shows **Active**.



Name	Type	Active	Autostart	VLAN a…	Ports/Slaves	Bond M…	CIDR
ens160	Network Device	Yes	Yes	No			192.168.77.74/24
ens32	Network Device	Yes	No	No			
vmbr0	Linux Bridge	Yes	Yes	No	ens32		10.1.149.74/24

The network adapter now shows to be active

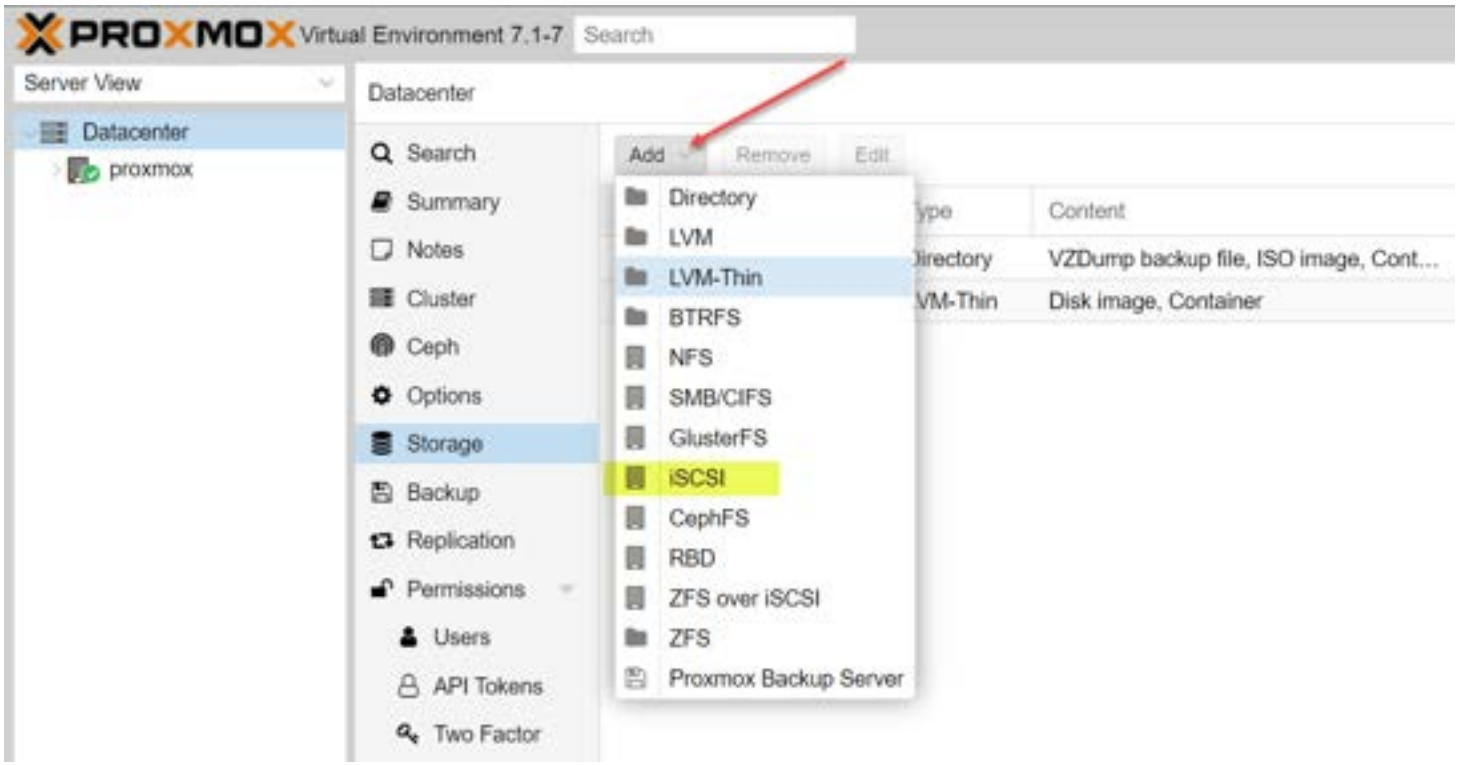
From your Proxmox server, ping your Synology iSCSI address to ensure you have connectivity.

```
Linux proxmox 5.13.19-2-pve #1 SMP PVE 5.13.19-4 (Mon, 29 Nov 2021 12:10:09 +0100) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Jan 18 21:39:12 CST 2022 on pts/0
root@proxmox:~# ping 192.168.77.7
PING 192.168.77.7 (192.168.77.7) 56(84) bytes of data.
64 bytes from 192.168.77.7: icmp_seq=1 ttl=64 time=0.373 ms
64 bytes from 192.168.77.7: icmp_seq=2 ttl=64 time=0.233 ms
64 bytes from 192.168.77.7: icmp_seq=3 ttl=64 time=0.274 ms
64 bytes from 192.168.77.7: icmp_seq=4 ttl=64 time=0.215 ms
64 bytes from 192.168.77.7: icmp_seq=5 ttl=64 time=0.258 ms
64 bytes from 192.168.77.7: icmp_seq=6 ttl=64 time=0.249 ms
64 bytes from 192.168.77.7: icmp_seq=7 ttl=64 time=0.256 ms
^C
--- 192.168.77.7 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6123ms
rtt min/avg/max/mdev = 0.215/0.265/0.373/0.047 ms
root@proxmox:~#
```

Verify you have connectivity to your iSCSI portal target of the Synology NAS

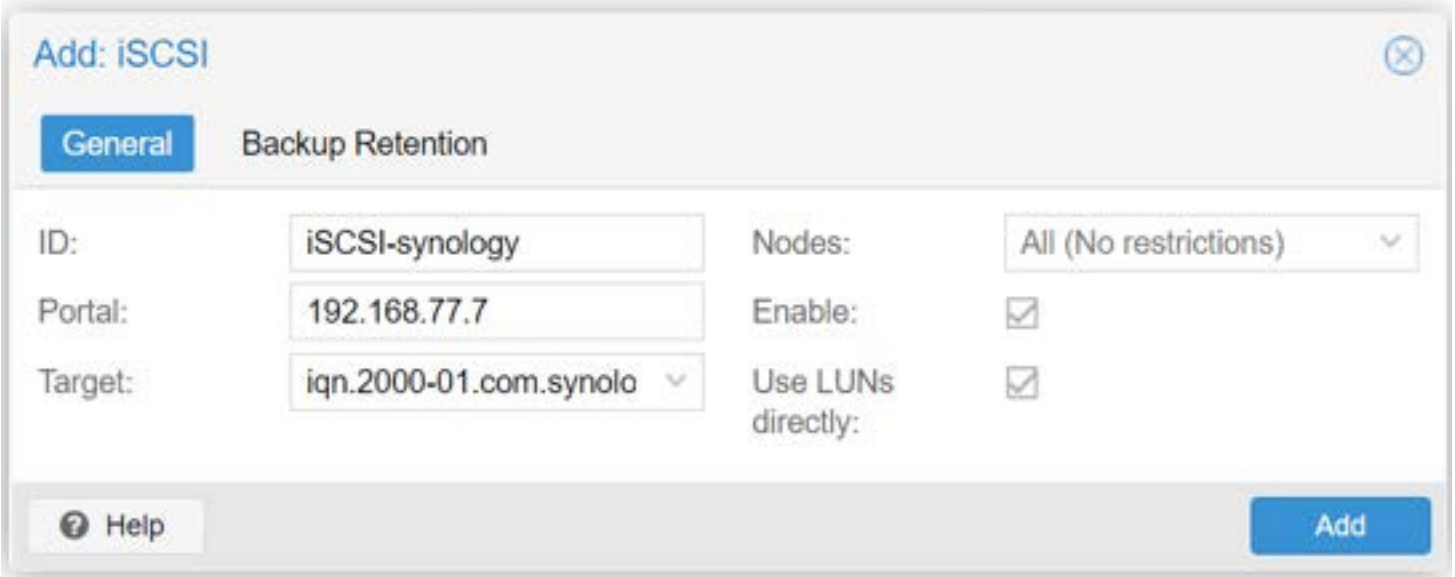
3. Add the iSCSI target to Proxmox

Next, lets add the Synology iSCSI target to Proxmox. Click your **Datacenter > Storage > Add**.



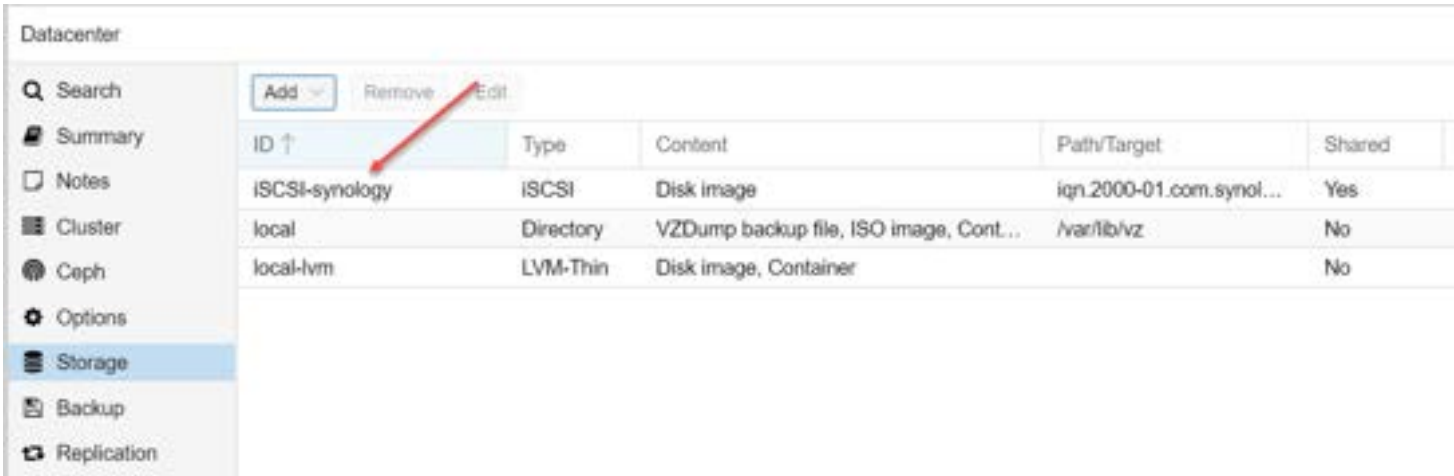
Add a new iSCSI target in Proxmox

Configure the iSCSI ID, Portal, and Target.



Fill in the configuration for your Synology iSCSI NAS target

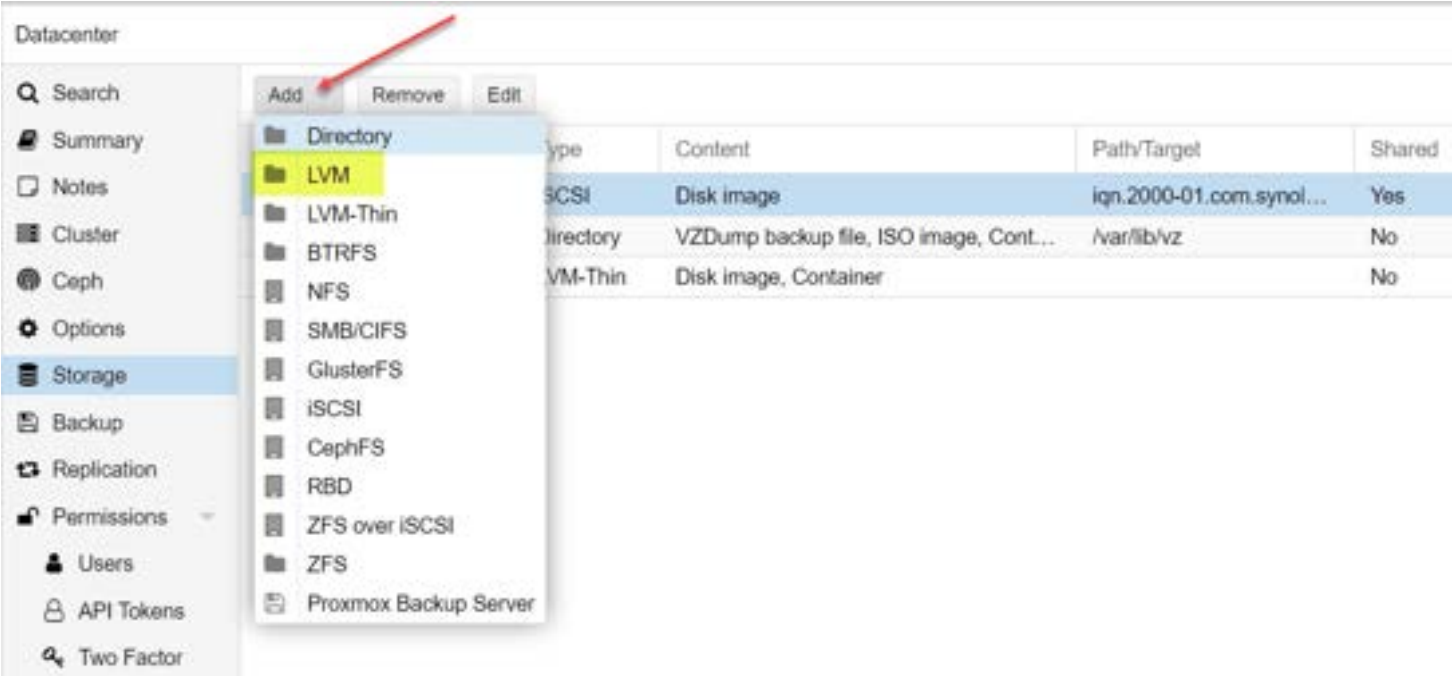
After adding the target, you will see it in your **Storage** list.



New iSCSI target has been added in Proxmox

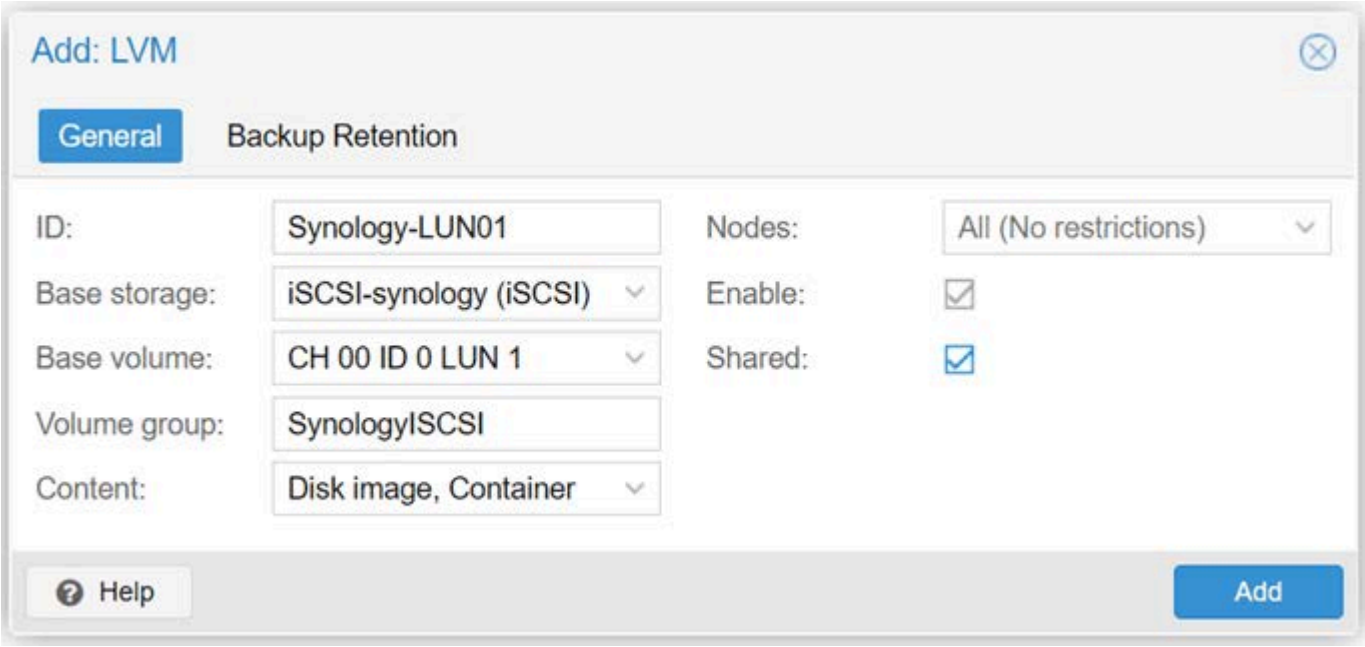
4. Create the LVM to the Synology iSCSI target

Now that we have the target added, we need to add an LVM to use the iSCSI storage. Click **Storage > Add > LVM**.



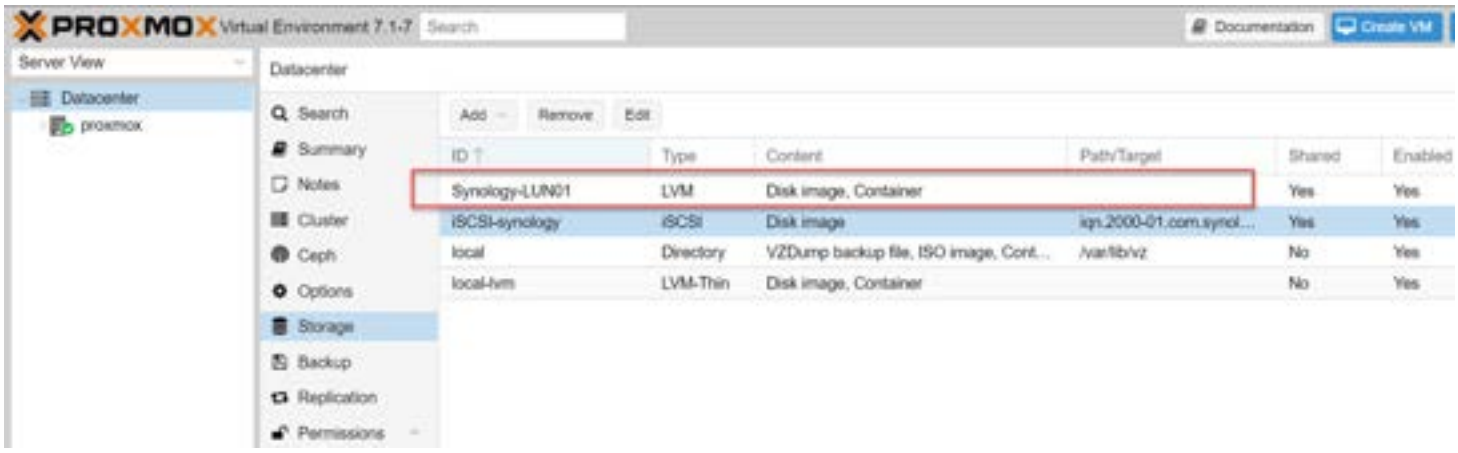
Add a new LVM in Proxmox

Add an ID, Base storage (choose from dropdown), Base volume (choose from dropdown), Volume Group (name this something intuitive), and Content as Disk image, Container.



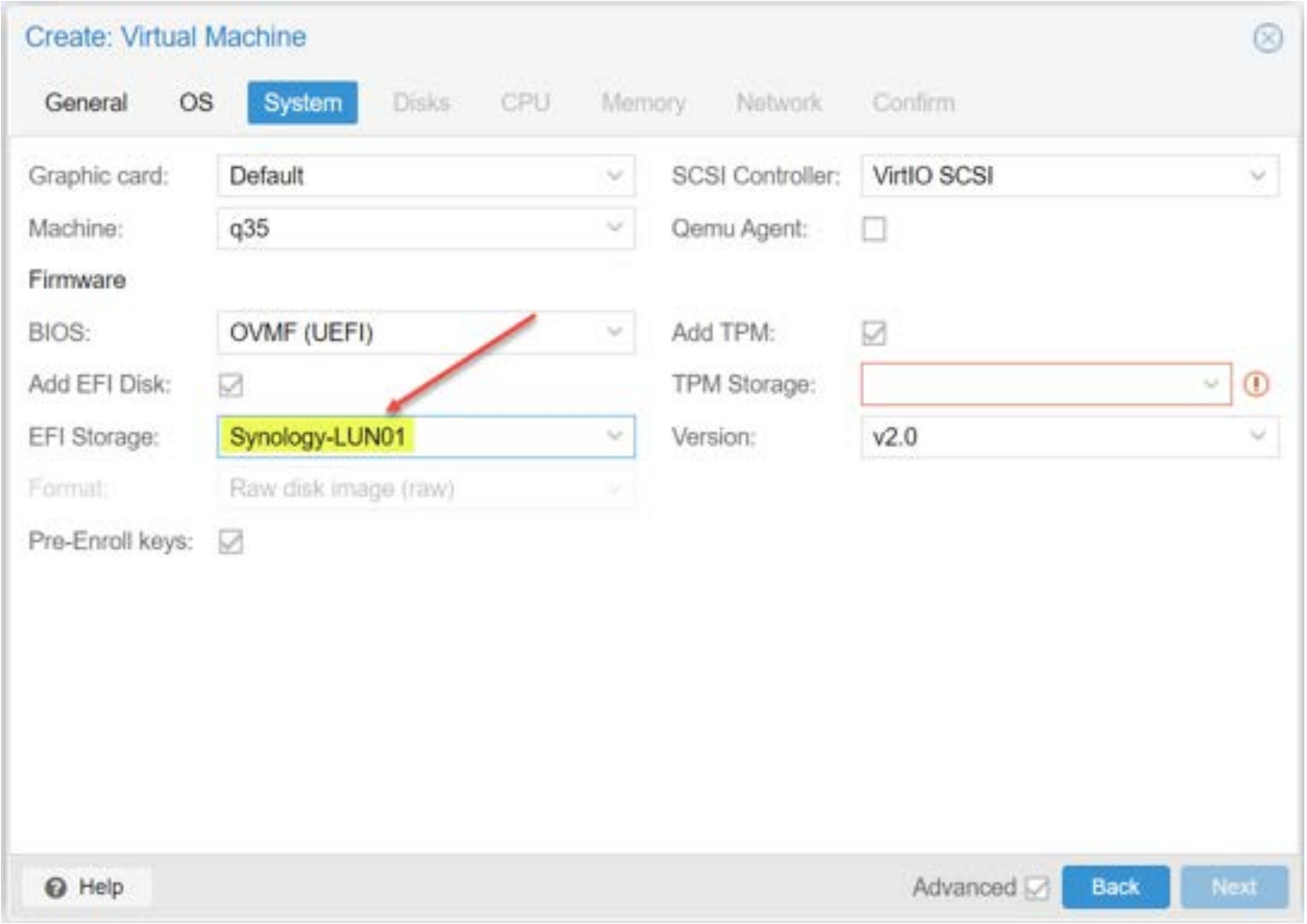
Configure the base storage pointed to the Synology iSCSI target

You will now see the new iSCSI LUN displayed in your list of storage.



New iSCSI LUN successfully added to Proxmox

Now, when you create a new Virtual Machine, you will see the iSCSI LUN listed as available to select.



Creating a new Proxmox virtual machine you can choose the Synology iSCSI LUN

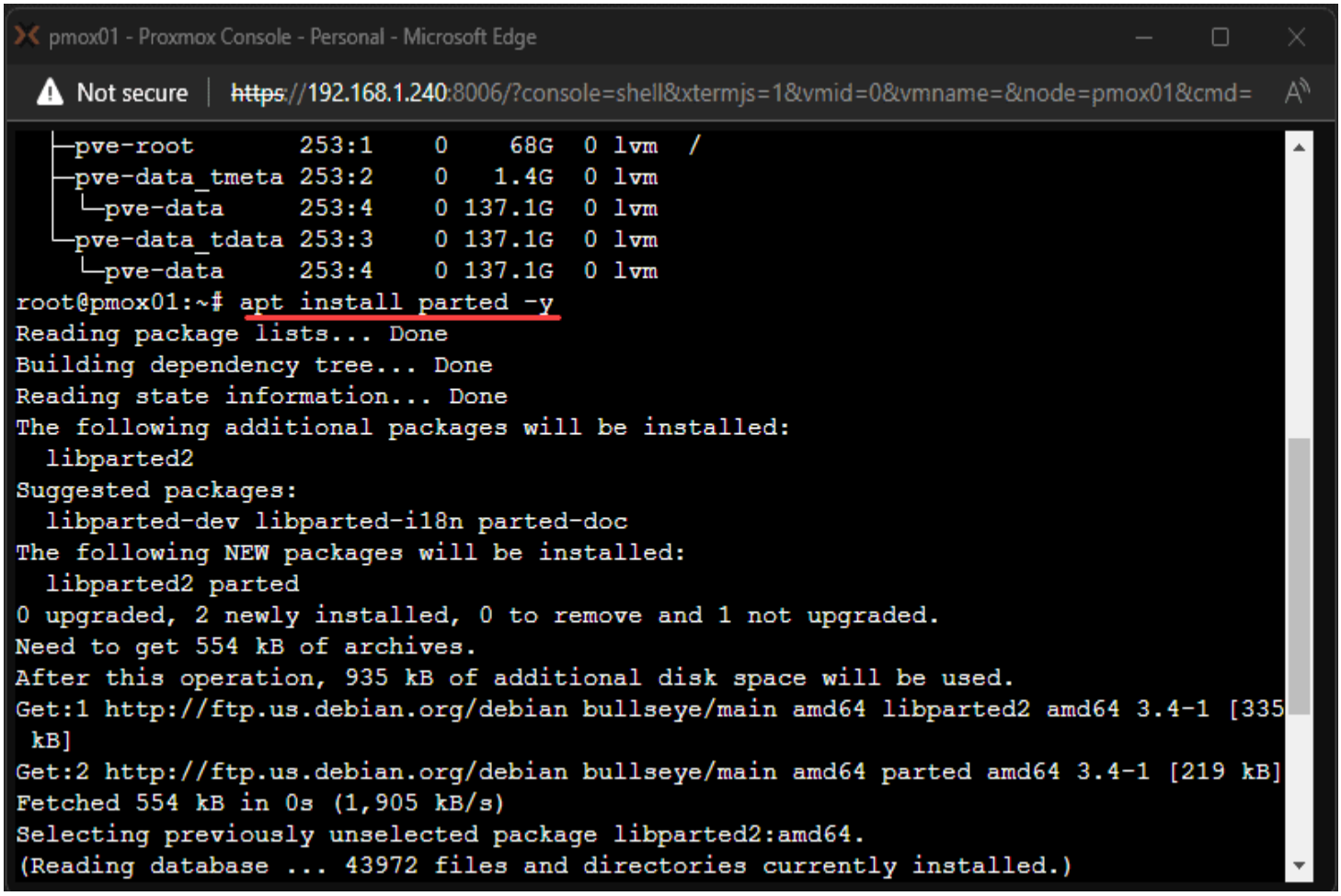
Wrapping Up

Hopefully, this quick walkthrough of setting up a Proxmox iSCSI target to Synology NAS helps to remove any uncertainty of how this is configured. From the Synology NAS side, the process is the same no matter which hypervisor you are using. Generally, the only change in how you add the iSCSI storage comes from the vendor side that you are adding the storage from. Using VMware vSphere and want to add an iSCSI target to your Synology NAS? Take a look at my post on how to do that here:

- iSCSI Synology VMware Configuration step-by-step

Proxmox add disk storage space – NVMe drive

April 10, 2023
Proxmox



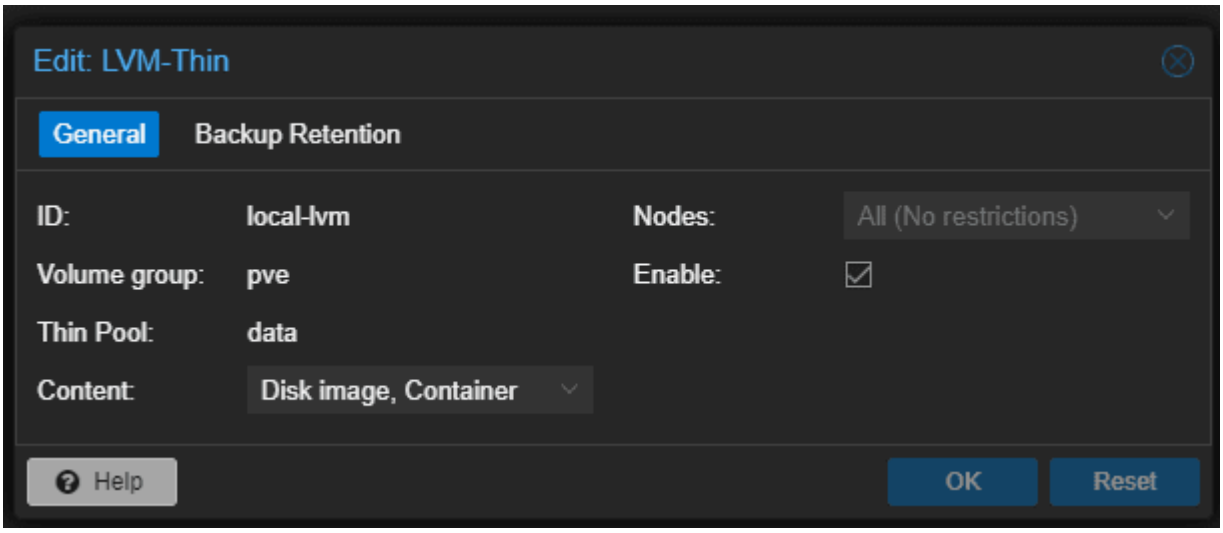
299d2808 4664 485d 9c31 efcf9675551c

Proxmox is a really great free and open-source virtualization platform that many are using in the home lab environment. However, one common challenge Proxmox users face is expanding storage space. With NVMe drives now being extremely cheap, they are a great choice for extra virtualization storage. Let's walk through the process of adding disk storage space to an NVMe drive in Proxmox, step by step.

Table of contents

Why add disk space to Proxmox?

You may need more hard disk space for a storage drive for virtual machines, containers, images, etc.



You may already have Proxmox installed on one disk and need to add storage to have more local storage. This process is as simple as adding a drive to Proxmox.

NVMe disks are cheap and great for adding speedy virtualization storage to your Proxmox host.

Add the physical drive to your Proxmox host

The first step is, of course, adding the new drive to your Proxmox host. Most current hosts have a hard drive M.2 slot for NVMe storage. Once added, we can begin the process to provision the storage.

Preparing the NVMe Drive

Before adding storage space to your Proxmox server, you must prepare the new disk. This involves installing necessary tools, creating a new partition table, and setting up the primary partition.

1. Install Parted:

To begin, you must install the Parted utility on your Proxmox server. Select Shell in the web interface to launch the command shell. This tool is used for manipulating block devices and creating partitions. To install Parted, run the following command:

```
apt install parted
```

pmox01 - Proxmox Console - Personal - Microsoft Edge

Not secure | https://192.168.1.240:8006/?console=shell&xtermjs=1&vmid=0&vmname=&node=pmox01&cmd=

```
  ├─pve-root       253:1    0    68G  0 lvm  /
  ├─pve-data_tmeta 253:2    0   1.4G  0 lvm
  │ └─pve-data     253:4    0 137.1G  0 lvm
  └─pve-data_tdata 253:3    0 137.1G  0 lvm
    └─pve-data     253:4    0 137.1G  0 lvm
root@pmox01:~# apt install parted -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libparted2
Suggested packages:
  libparted-dev libparted-i18n parted-doc
The following NEW packages will be installed:
  libparted2 parted
0 upgraded, 2 newly installed, 0 to remove and 1 not upgraded.
Need to get 554 kB of archives.
After this operation, 935 kB of additional disk space will be used.
Get:1 http://ftp.us.debian.org/debian bullseye/main amd64 libparted2 amd64 3.4-1 [335
 kB]
Get:2 http://ftp.us.debian.org/debian bullseye/main amd64 parted amd64 3.4-1 [219 kB]
Fetched 554 kB in 0s (1,905 kB/s)
Selecting previously unselected package libparted2:amd64.
(Reading database ... 43972 files and directories currently installed.)
```

1. List block devices:

Next, use the **lsblk** command to identify the new NVMe drive you want to add to your Proxmox installation. For example, the drive may appear as **dev/nvme0n1**. You will see the normal disks such as dev sda dev sda1.

pmox01 - Proxmox Console - Personal - Microsoft Edge

Not secure | https://192.168.1.240:8006/?console=shell&xtermjs=1&vmid=0&vmname=&node=pmox01&cmd=

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Apr 10 15:46:49 CDT 2023 on tty1
root@pmox01:~# lsblk
NAME                MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda                   8:0    1  58.9G  0 disk
├─sda1                8:1    1   224K  0 part
├─sda2                8:2    1   2.8M  0 part
├─sda3                8:3    1     1G  0 part
└─sda4                8:4    1   300K  0 part
nvme0n1             259:0    0 931.5G  0 disk
├─nvme0n1p1         259:2    0     2M  0 part
└─nvme0n1p2         259:3    0 931.5G  0 part
nvme1n1             259:1    0 232.9G  0 disk
├─nvme1n1p1         259:4    0  1007K  0 part
├─nvme1n1p2         259:5    0     1G  0 part
└─nvme1n1p3         259:6    0 231.9G  0 part
  ├─pve-swap        253:0    0     8G  0 lvm  [SWAP]
  ├─pve-root        253:1    0    68G  0 lvm  /
  ├─pve-data_tmeta  253:2    0   1.4G  0 lvm
  │ └─pve-data      253:4    0 137.1G  0 lvm
  └─pve-data_tdata  253:3    0 137.1G  0 lvm
    └─pve-data      253:4    0 137.1G  0 lvm
root@pmox01:~#
```

1. Create a new partition table:

Once you have identified the new disk, you can create a new partition table. Run the following command to create a type GPT partition table on the NVMe drive:

```
parted /dev/nvme0n1 mklabel gpt
```

```
root@pmox01:~# ^C
root@pmox01:~# parted /dev/nvme0n1 mklabel gpt
Warning: The existing disk label on /dev/nvme0n1 will be destroyed and all data on
this disk will be lost. Do you want to continue?
Yes/No? Yes
Information: You may need to update /etc/fstab.

root@pmox01:~#
```

Creating the Primary Partition

After you have prepared the new disk, the next step is to create the primary partition on the NVMe drive.

1. Create a primary partition

To create the primary partition, run the following command:

```
parted /dev/nvme0n1 mkpart primary ext4 0% 100%
```

```
root@pmox01:~# parted /dev/nvme0n1 mkpart primary ext4 0% 100%
Information: You may need to update /etc/fstab.
```

This command creates a new primary ext4 partition that spans the entire NVMe drive and will remove existing partitions. So be careful and ensure this is what you would like to do.

1. Format the partition:

Next, format the new partition with the ext4 filesystem:

```
mkfs.ext4 /dev/nvme0n1p1
```

1. Name the storage for your data

You can name the storage something intuitive using the command:

```
mkfs.ext4 -L vmstorage /dev/nvme0n1
```

```
root@pmox01:~# mkfs.ext4 -L vmstorage /dev/nvme0n1
mke2fs 1.46.5 (30-Dec-2021)
Found a gpt partition table in /dev/nvme0n1
Proceed anyway? (y,N) y
Discarding device blocks: done
Creating filesystem with 244190646 4k blocks and 61054976 inodes
Filesystem UUID: 61cf39ed-a308-4eae-a9d2-a019eaaa1697
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
        102400000, 214990848

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting information: done

root@pmox01:~#
```

Mounting the New Partition

With the primary partition created, you must now mount it to a mount point on your Proxmox server.

1. Create a new directory:

First, create a new directory to serve as the mount point for the new partition. For example:

```
mkdir /mnt/vmstorage
```

1. Edit the /etc/fstab file:

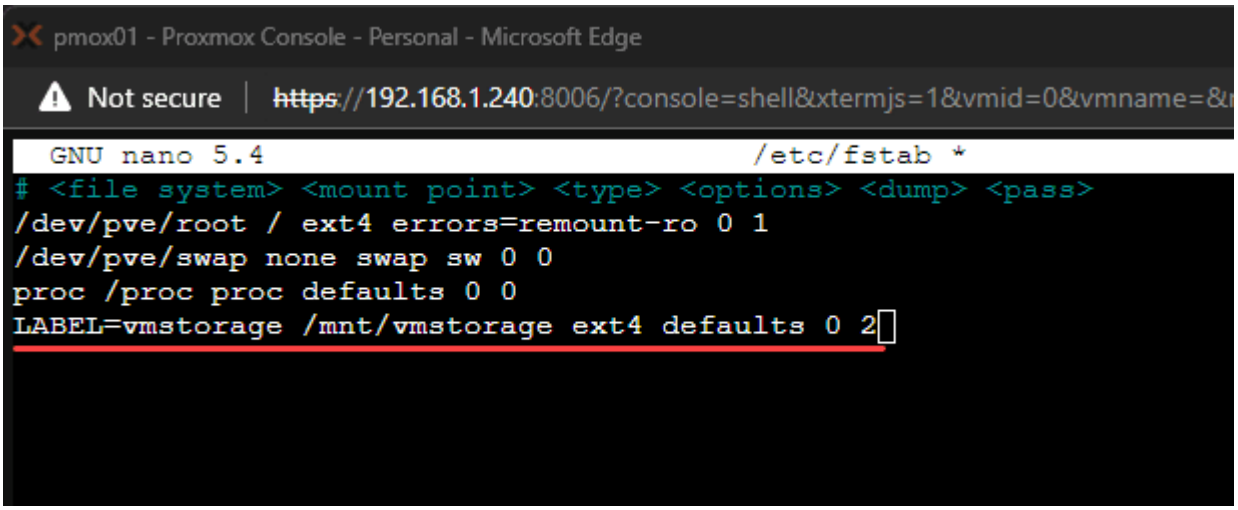
Next, edit the /etc/fstab file to ensure the new partition will auto mount upon reboot. Open the file with a text editor like Nano:

```
nano /etc/fstab
```

```
root@pmox01:~# mkdir -p /mnt/vmstorage
root@pmox01:~# nano /etc/fstab
root@pmox01:~#
```

Add the following line to the file, replacing **/dev/nvme0n1p1** with the appropriate device identifier for your NVMe drive:

```
LABEL=vmstorage /mnt/vmstorage ext4 defaults 0 2
```



Save the changes and exit the text editor.

1. Mount the new partition:

Finally, mount the new partition to the mount point:

```
mount /mnt/vmstorage
```

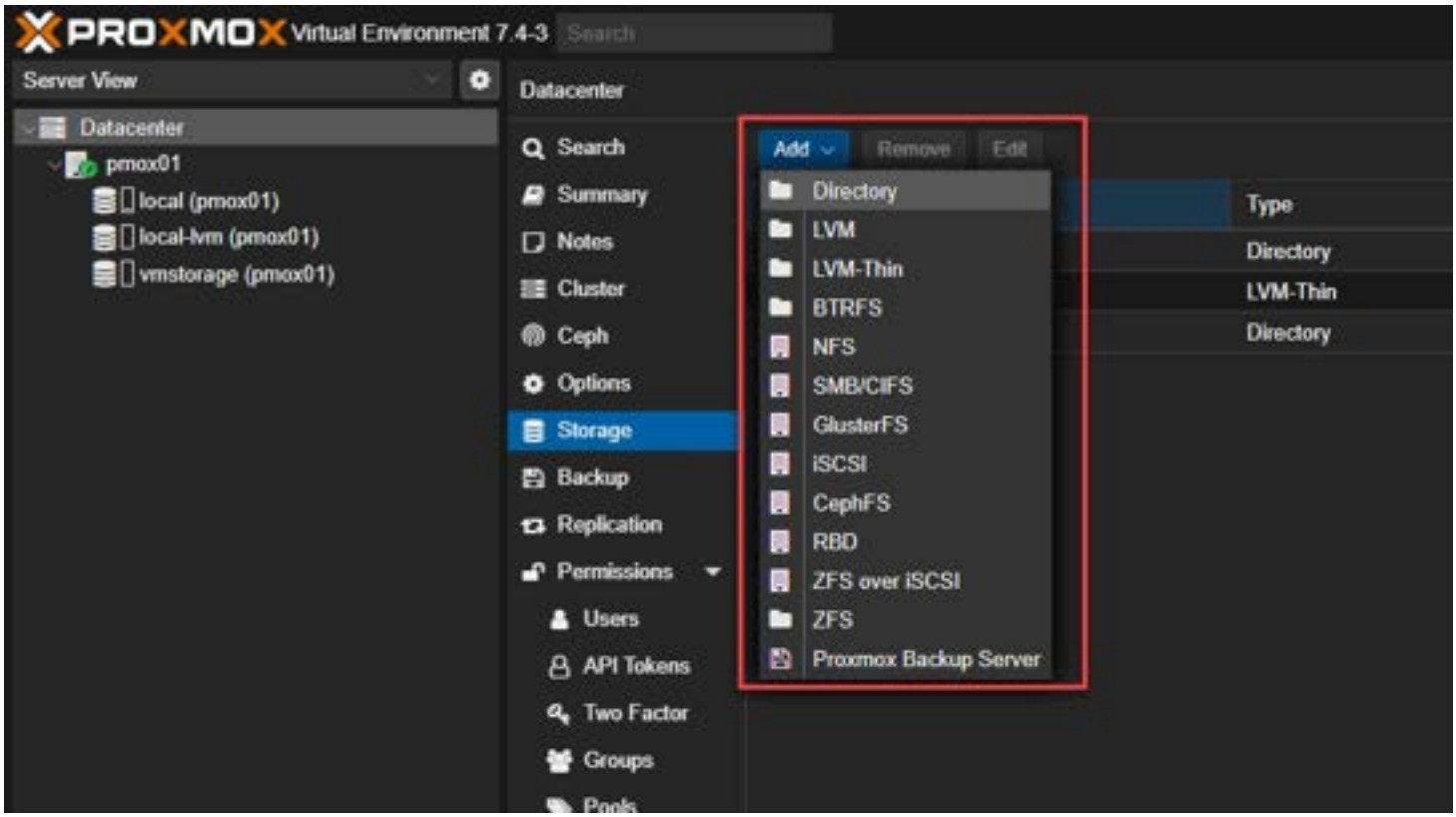
Adding Storage Space to Proxmox

Now that the new partition is mounted, you can add it as storage space in the Proxmox web interface.

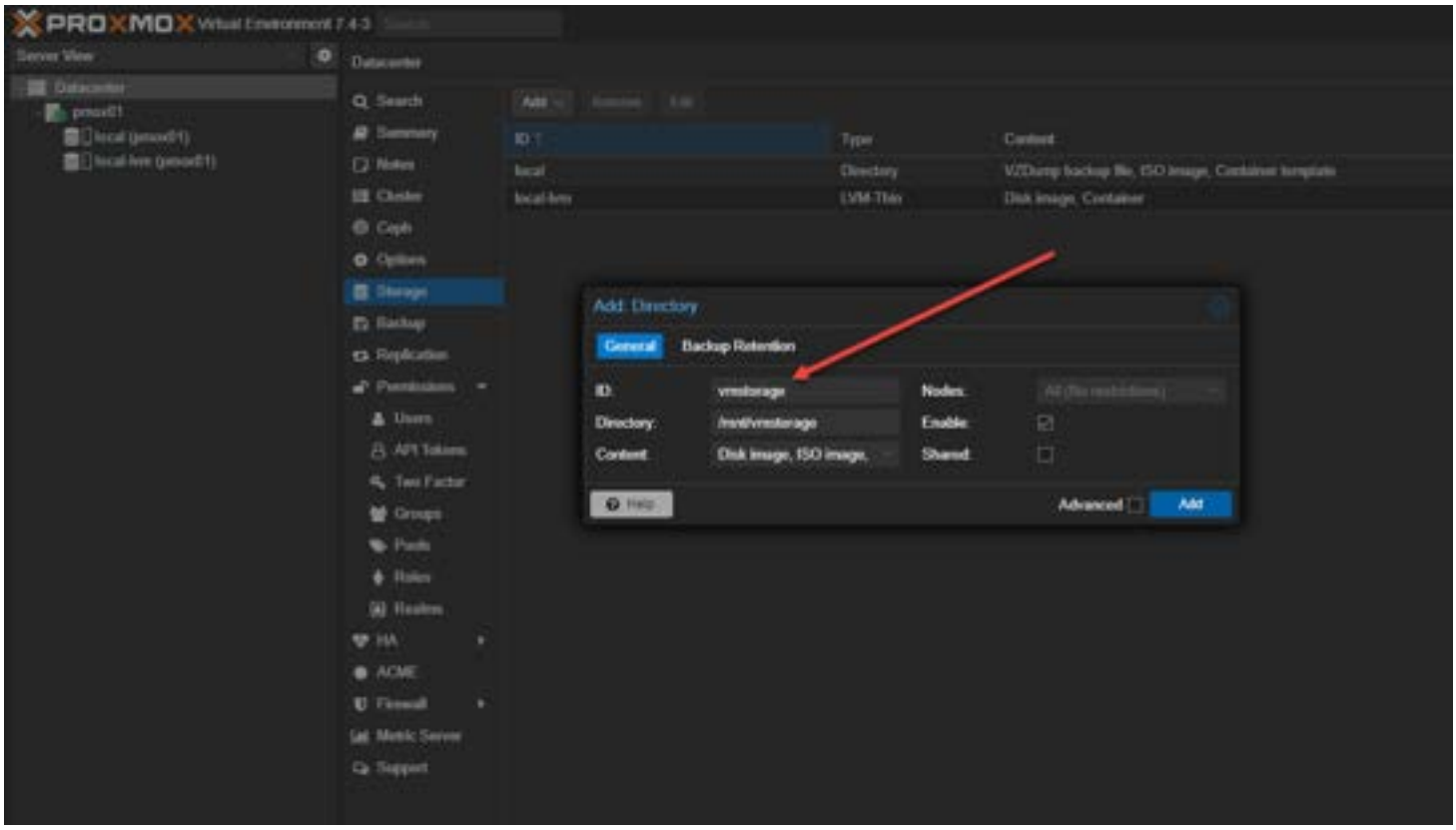
1. Log in to the Proxmox web interface.

2. Navigate to the “Datacenter” tab and click on “Storage.”

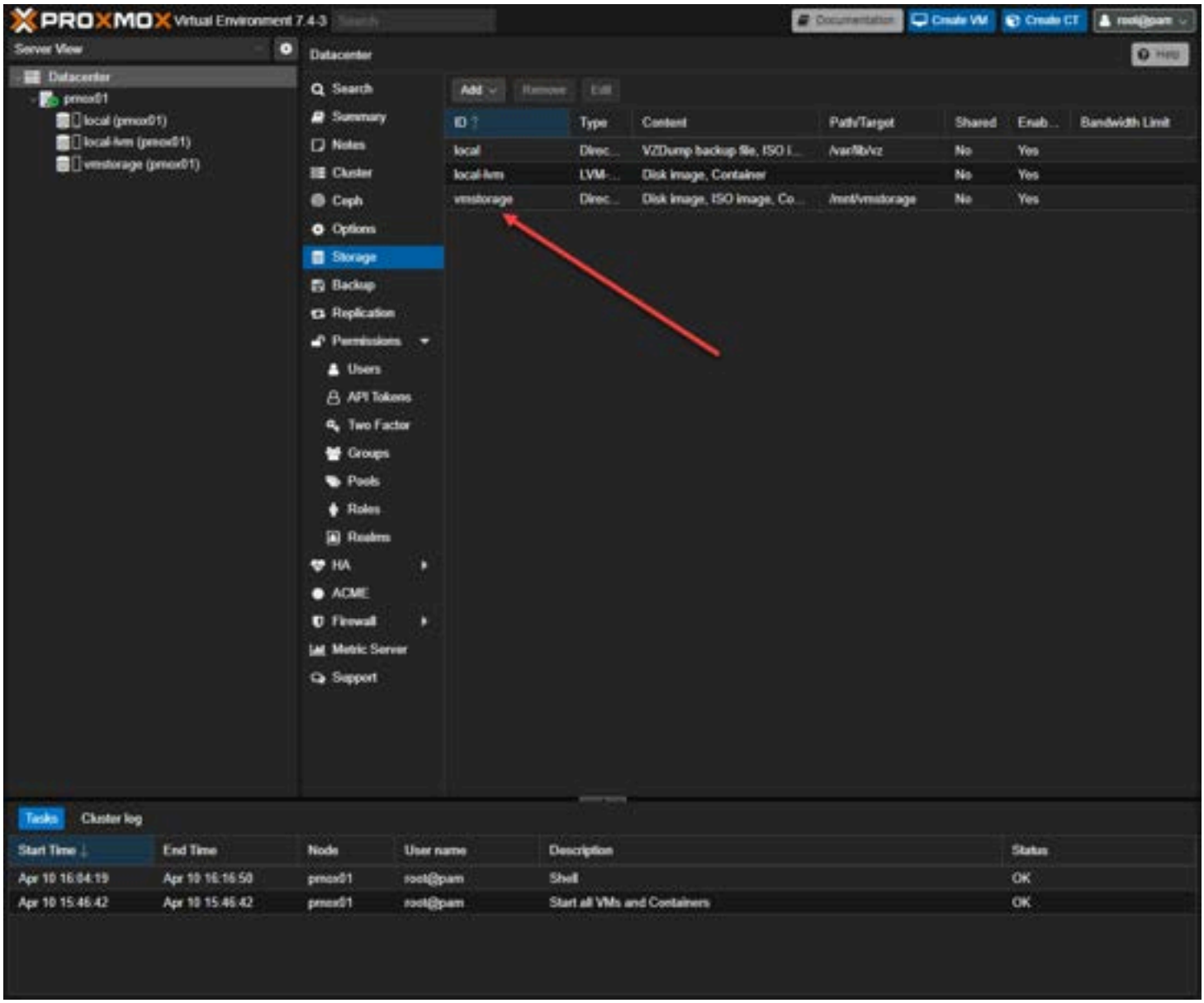
3. Click on the “Add” button and select “Directory” from the dropdown menu. There will be several options including lvm thin think provisioning, volume group, thin provisioning.



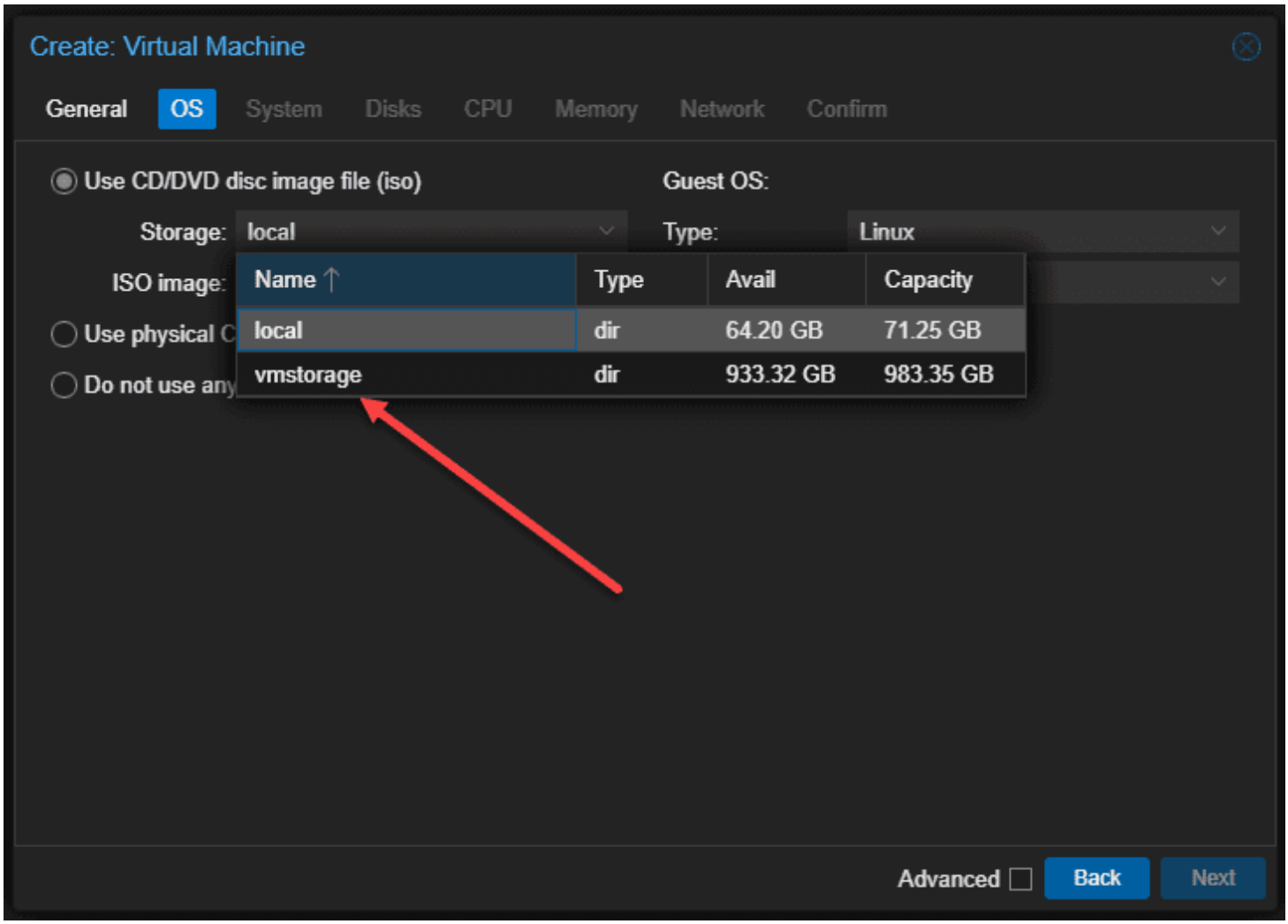
4. In the “Add Directory” window, enter a unique ID for the new storage, and set the “Directory” field to the mount point you created earlier (e.g., **/mnt/nvme-storage**). Choose the appropriate “Content” types, such as “select Disk image,” “Container template,” or “Backup.” Click “Add” to save the configuration.



5. The new storage space will now be available in the Proxmox web interface for virtual machines and containers.



Now, when you create a new virtual machine or container you will see the storage available to select in the storage drop down.



Performing Backups to the New Storage

With the new storage space added to Proxmox, you can also use it as backup storage for your virtual machines and containers. To create a backup, follow these steps:

1. Select the virtual machine or container you want to back up in the Proxmox web interface.

2. Click on the “Backup” button in the top menu.

3. In the “Backup” window, choose the new storage space as the “Storage” target and configure the other backup options as needed. Click “Backup” to initiate the process.

4. Monitor the backup progress in the “Task Log” tab.

Related posts

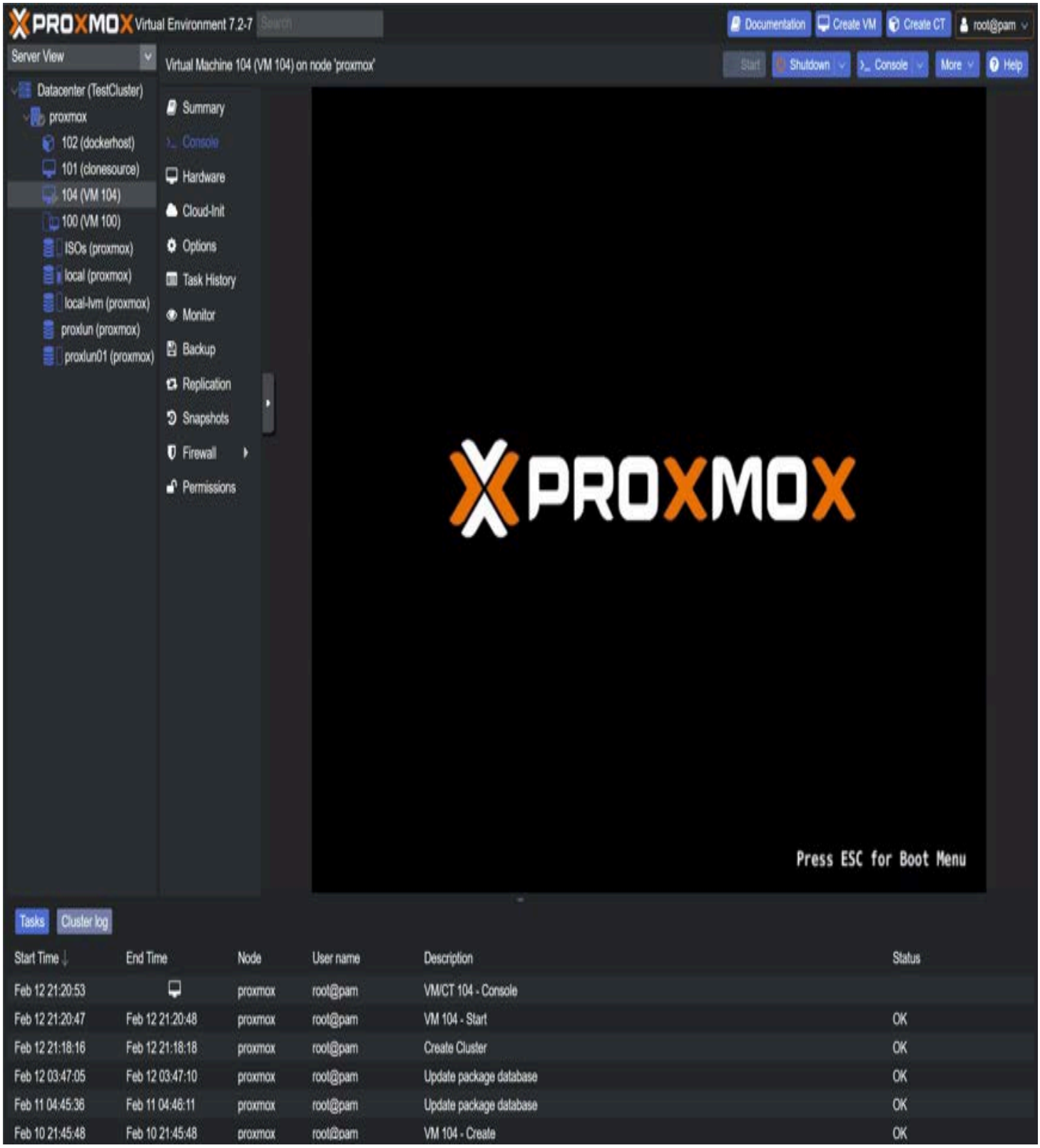
- Nested Proxmox VMware installation in ESXi
- Proxmox Create ISO Storage Location – disk space error
- Proxmox cluster installation and configuration
- Proxmox firewall rules configuration
- Proxmox Update No Subscription Repository Configuration

Wrapping up

Proxmox is a great solution that is free, open-source, and incorporates many great features in the platform. Adding storage is fairly straightforward, but does involve a few steps from the command shell to mount the storage, format the disk, and add it to the system as available storage for virtual machines and containers.

Proxmox cluster installation and configuration

February 13, 2023
Proxmox



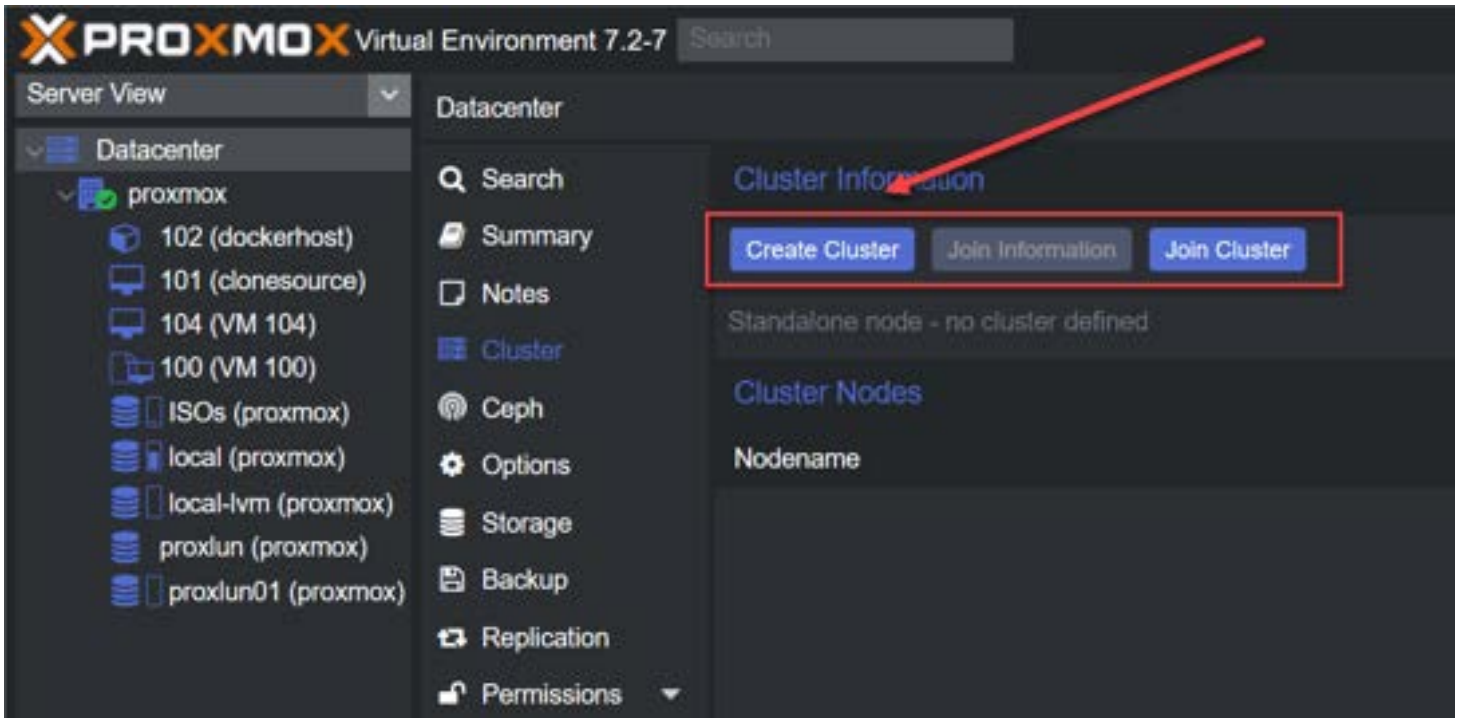
9adb5010 cd8d 4fe7 891d 9c52d62fc3f2

Proxmox Cluster is a group of physical servers that work together to provide a virtual environment for creating and managing virtual machines and other resources. In this blog post, we will go over the steps to build a Proxmox Cluster and the benefits it provides.

What is Proxmox Cluster?

Proxmox Cluster is a group of physical servers that work together to provide a virtual environment for creating and managing virtual machines and other resources.

The Proxmox Cluster uses the Proxmox Virtual Environment (VE) to provide a virtual environment for creating and managing virtual machines.

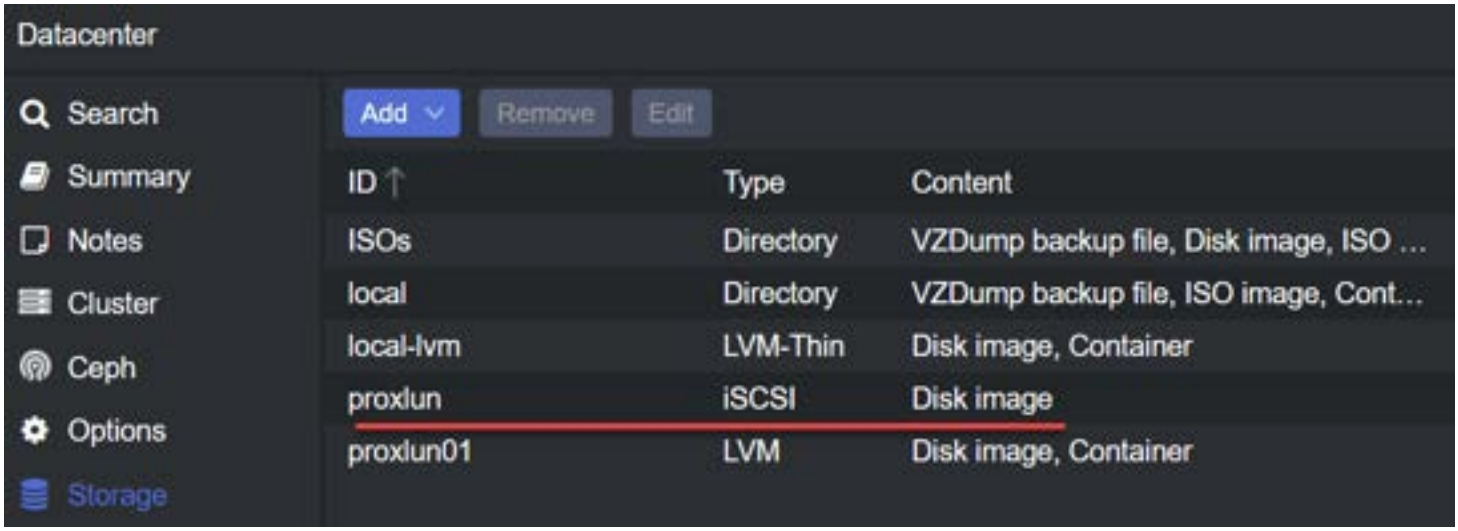


Minimum Requirements for Building a Proxmox Cluster

To build a Proxmox Cluster, you will need at least two Proxmox servers, or nodes for a VE cluster. It is recommended to use identical hardware for all nodes in the cluster to ensure compatibility and ease of management.

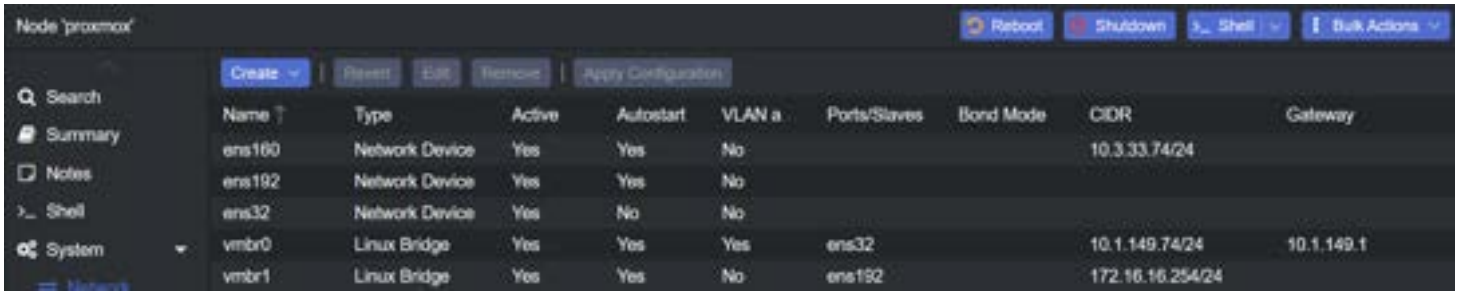
The Proxmox servers will communicate with each other to perform management tasks and ensure your virtual environment's reliability.

Having shared storage is a good idea as this will allow the most seamless and best configuration for production workloads. It allows workloads to be brought back up quickly if one host fails.



Importance of IP Addresses in Proxmox Cluster

Each node in a Proxmox Cluster must have a unique IP address. The IP addresses are used for cluster communication and to identify each node in the cluster. It is important to make sure that each node has a unique IP address and that the addresses are reachable from other nodes in the network.



Storing Configuration Files for Proxmox Cluster

The configuration files for a Proxmox Cluster can be stored on either local storage directly attached to a node or shared storage accessible from multiple nodes.

Choosing the appropriate storage option for your cluster is important based on your needs and the resources available.

Setting Up Cluster Communication

The corosync communication protocol manages communication between nodes in a Proxmox Cluster. The protocol is responsible for ensuring that nodes in the cluster can communicate with each other and for managing the transfer of information between nodes.

Modifying the Configuration File

The configuration file for a Proxmox Cluster includes the settings for the corosync communication protocol, the cluster manager, and the virtual environment.

The configuration file is stored in a database-driven file system and can be easily modified to meet the needs of your virtual environment.

Handling Failed Nodes in a Proxmox Cluster

In the event of a failed node in a Proxmox Cluster, the remaining node will continue to function normally and ensure your virtual environment's reliability. The cluster manager is responsible for automatically failing over to the remaining nodes in the event of a failure.

The Role of the Cluster Manager

The cluster manager is responsible for performing management tasks in a Proxmox Cluster, such as live migrations of virtual machines and automatic failover in case of a failed node. The cluster manager is an integral component of a Proxmox Cluster and ensures that the virtual environment remains up and running even in the event of a failure.

Benefits of a Proxmox Cluster

A Proxmox Cluster provides many benefits, including high availability, easy migration of virtual machines, and automatic failover in case of a failed node. With a Proxmox Cluster, you can ensure that your virtual environment is always up and running and that your virtual machines are always available to users.

Easy Migration of Virtual Machines

Another benefit of a Proxmox Cluster is easy migration of virtual machines. With a Proxmox Cluster, you can easily migrate virtual machines from one node to another, providing flexibility and ease of management.

Node Types in a Proxmox Cluster

In a Proxmox Cluster, there are two types of nodes: the main node and the slave node or second node. The main node is responsible for performing management tasks, while the slave node is responsible for running virtual machines.

In the event of a failure of the main node, the slave node will take over and perform management tasks until the main node is restored.

Grouping Nodes in a Proxmox Cluster

In a Proxmox Cluster, nodes can be grouped together to provide additional functionality and ease of management. For example, you can group nodes together based on their location or based on the virtual machines they are running. This grouping of nodes allows you to manage and monitor your virtual environment easily.

Proxmox Single Node Clusters

In addition to multi-node clusters, Proxmox also supports single-node clusters. A single-node cluster is a Proxmox cluster that consists of only one node and is typically used for smaller virtual environments or for testing and development purposes.

A single-node cluster in Proxmox provides many of the benefits of a multi-node cluster, such as creating and managing virtual machines and using local storage for virtual machine storage.

Additionally, a single node cluster provides a simple and easy-to-use virtual environment well-suited for small or simple virtual environments.

To set up a single-node cluster in Proxmox, you will need to install Proxmox on a single node and configure the network settings. Once Proxmox is installed, you can create a new single node cluster using the Proxmox Web GUI or the command line.

When creating a single node cluster, properly configuring the firewall ensures the virtual environment is secure. Additionally, it is important to plan properly and backup the virtual machines and configurations to ensure the reliability of the virtual environment.

Live Migration of Virtual Machines

Live migration is a feature in a Proxmox Cluster that allows you to move virtual machines from one node to another without any downtime. This feature is useful for performing maintenance tasks on a node or for balancing the load between nodes in the cluster.

High Availability in Proxmox Cluster

High availability is a key benefit of a Proxmox Cluster. With high availability, you can ensure that your virtual environment remains up and running even in a failure.

The cluster manager is responsible for automatically failing over to the remaining nodes in the event of a failure, ensuring that your virtual environment remains up and running.

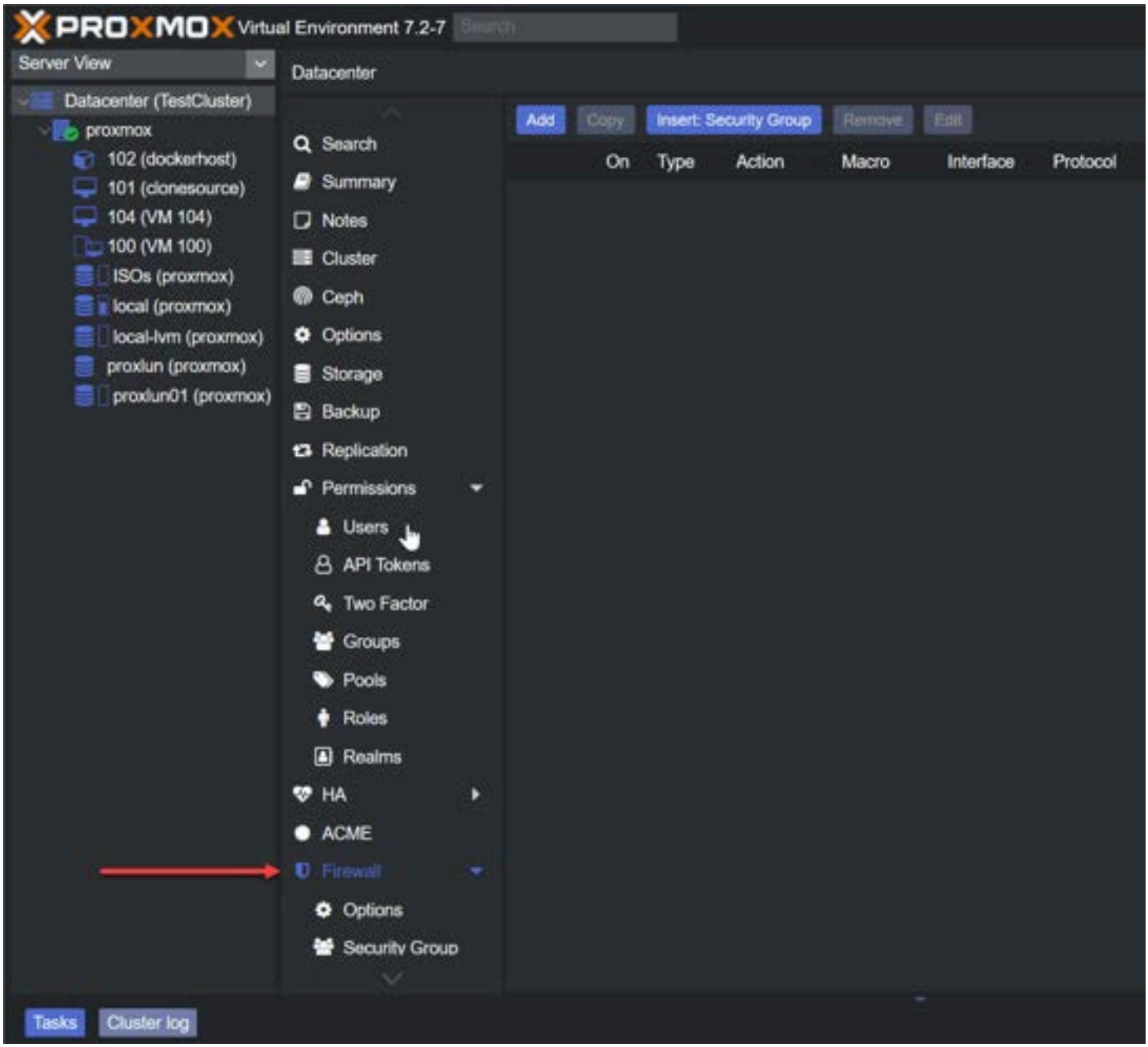
Considerations for Building a Proxmox Cluster

When building a Proxmox Cluster, there are several important considerations to keep in mind. These include the hardware requirements, the network requirements, and the firewall requirements.

It is important to thoroughly research and plan your Proxmox Cluster to ensure that it meets your needs and provides the desired level of reliability.

Firewall Requirements

When building a Proxmox Cluster, it is important to consider the firewall requirements. The Proxmox Cluster uses the TCP port to communicate between nodes, and it is important to ensure that this port is open on the firewall.



Additionally, it is important to consider any security requirements and to properly configure the firewall to meet these requirements.

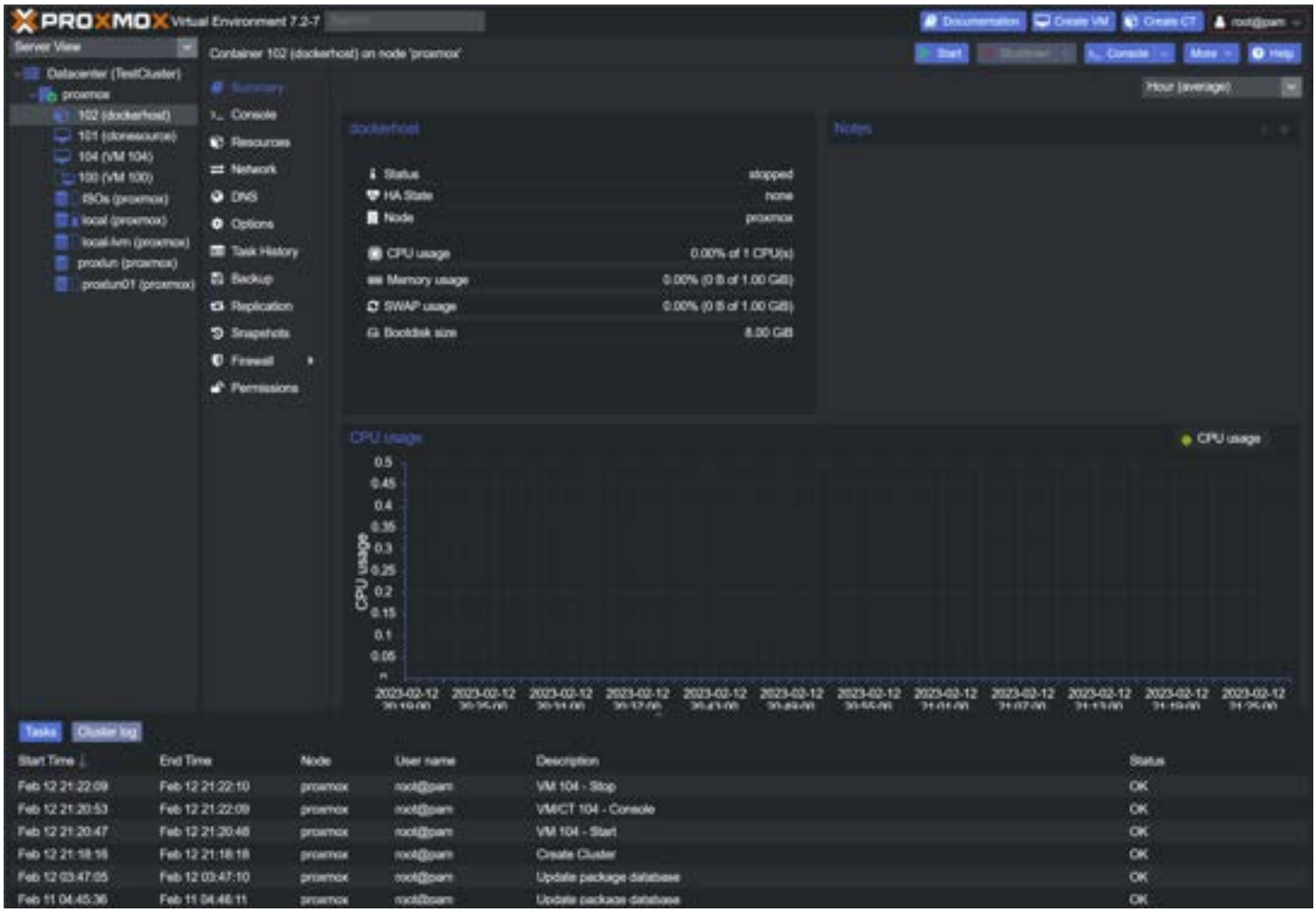
Home Lab Environments

Proxmox Clusters are not just for large data centers and enterprise environments. They can also be used in home lab environments to provide a virtual environment for testing and learning purposes.

A home lab environment typically consists of a small number of physical servers, often only one or two, and is used for testing and learning purposes.

With a Proxmox Cluster in a home lab environment, you can experience the benefits of a virtual environment, such as high availability and easy migration of virtual machines, without the need for a large number of physical servers.

When setting up a Proxmox Cluster in a home lab environment, it is important to consider the hardware requirements and choose hardware compatible with the Proxmox software.



Additionally, it is important to consider the network requirements and properly configure the firewall to ensure the cluster can communicate with other nodes.

It is also important to properly secure the Proxmox Cluster in a home lab environment. This includes securing the root password and properly configuring the firewall to prevent unauthorized access.

Proxmox Clusters in home lab environments provide a great opportunity to learn about virtual environments and to gain hands-on experience with Proxmox. With a Proxmox Cluster in a home lab environment, you can explore the features and benefits of a virtual environment and develop the skills you need to effectively manage virtual environments in real-world environments.

Creating a Proxmox Cluster with the CLI step-by-step

Step 1: Install Proxmox on each node

The first step in setting up a Proxmox Cluster is to install Proxmox on each node. To do this, you must download the Proxmox ISO file and create a bootable USB drive. Once the USB drive is created, you can boot each node from the USB drive and follow the prompts to install Proxmox.

Step 2: Configure the network settings

Once Proxmox is installed on each node, you must configure the network settings. This includes assigning a unique IP address to each node and configuring the firewall to allow communication between nodes.

Step 3: Create a new cluster

To create a new Proxmox Cluster, you will need to use the following command on one of the nodes:

```
pvecm create <cluster-name>
```

This command will create a new cluster with the specified name and make the node the main node.

Step 4: Join additional nodes to the cluster

To join additional nodes to the cluster, you will need to use the following join cluster command on each node:

```
pvecm join <ip-address-of-the-main-node>
```

This command will provide the necessary information to join the cluster, including the IP address of the main node and the cluster communication port.

Step 5: Configure the corosync communication protocol

The corosync communication protocol is used to manage communication between nodes in a Proxmox Cluster. To configure the corosync communication protocol, you will need to modify the configuration file for the cluster.

This file is stored in a database-driven file system and can be easily modified to meet the needs of your virtual environment.

Step 6: Add virtual machines to the cluster

Once the Proxmox Cluster is set up, you can add virtual machines. To do this, you must use the Proxmox Web GUI to create and configure virtual machines.

The virtual machines can be easily migrated between nodes in the cluster, providing flexibility and ease of management.

Step 7: Monitor and maintain the cluster

To ensure the reliability of your virtual environment, it is important to monitor and maintain your Proxmox Cluster. This includes monitoring the status of the nodes in the cluster, performing regular maintenance tasks, and updating the cluster

software as needed.

Create a Proxmox cluster using the Web GUI

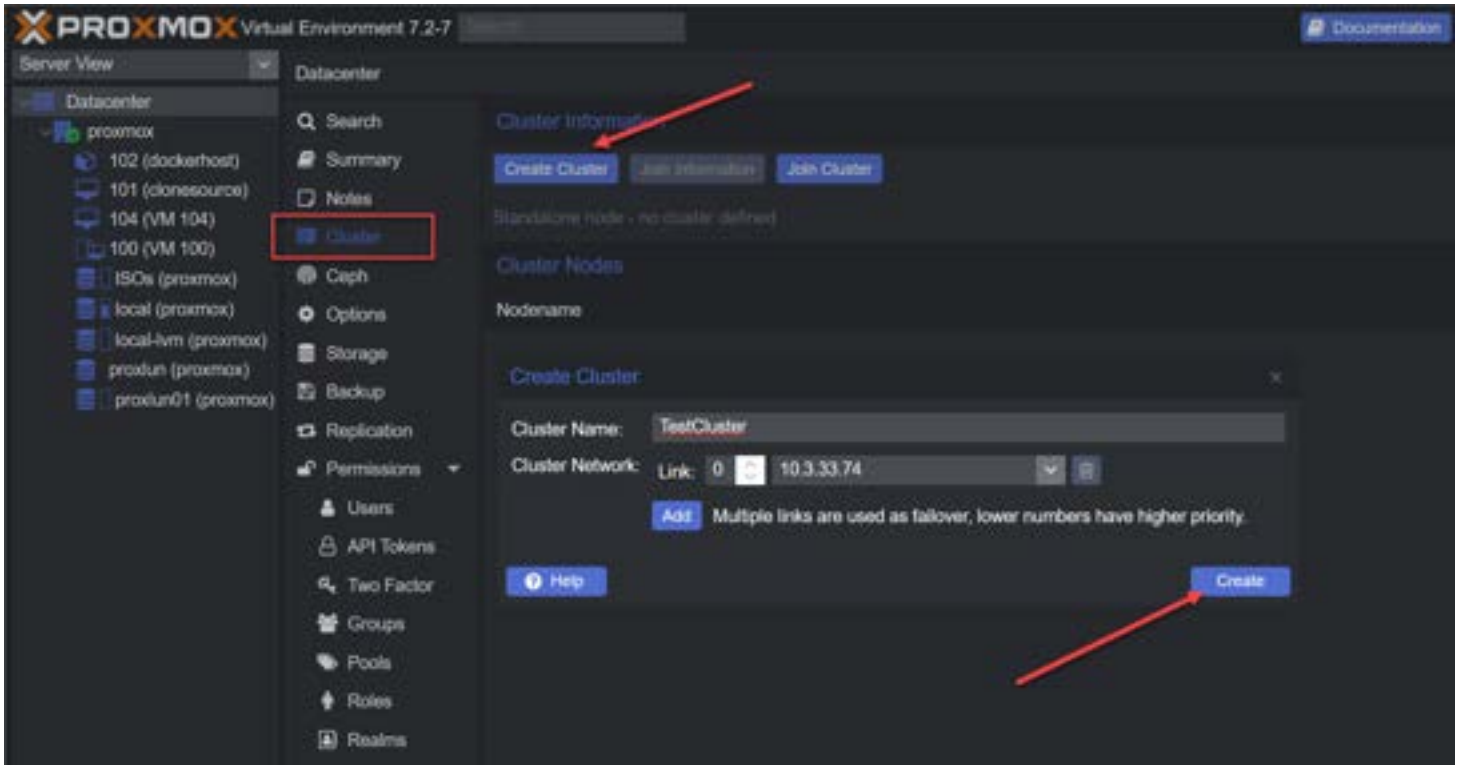
Step 1: Log in to the Proxmox Web GUI

To create a Proxmox Cluster using the Proxmox Web GUI, you will need to log in to the Proxmox Web GUI on one of the nodes in the cluster.

The Proxmox Web GUI can be accessed by navigating to **https://<node-ip-address>:8006** in a web browser.

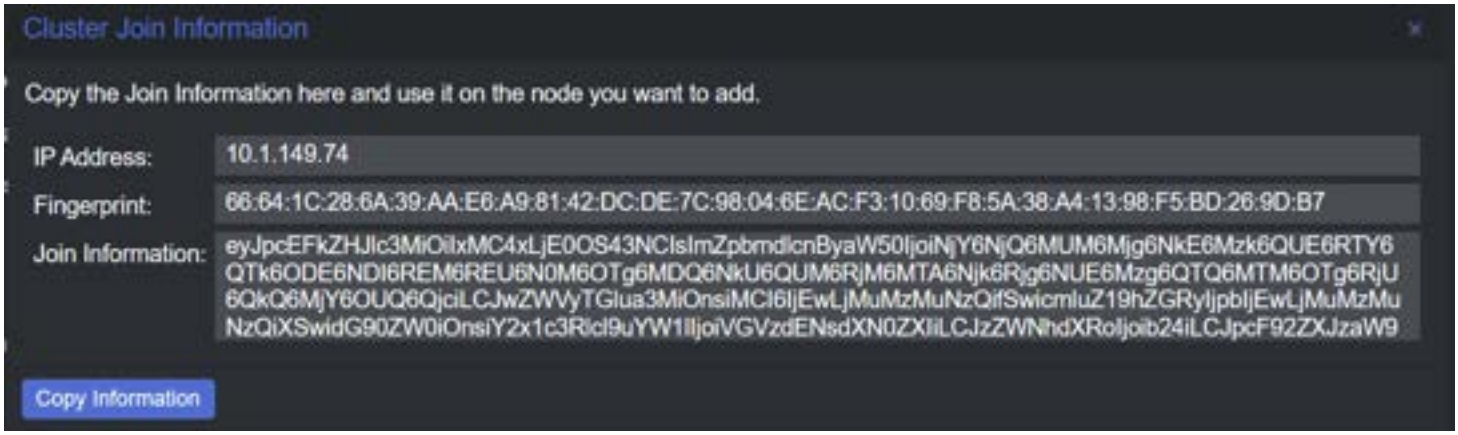
Step 2: Create a new cluster

To create a new cluster, click on the “Cluster” tab in the Proxmox Web GUI and then click on the “Create Cluster” button. This will open a dialog where you can enter the name of the new cluster.



Step 3: Add nodes to the cluster

Once the new cluster has been created, you can add additional nodes to the cluster. To do this, click on the “Cluster” tab in the Proxmox Web GUI and then click on the “Add Node” button. This will open a dialog where you can enter the node’s IP address you want to add to the cluster.

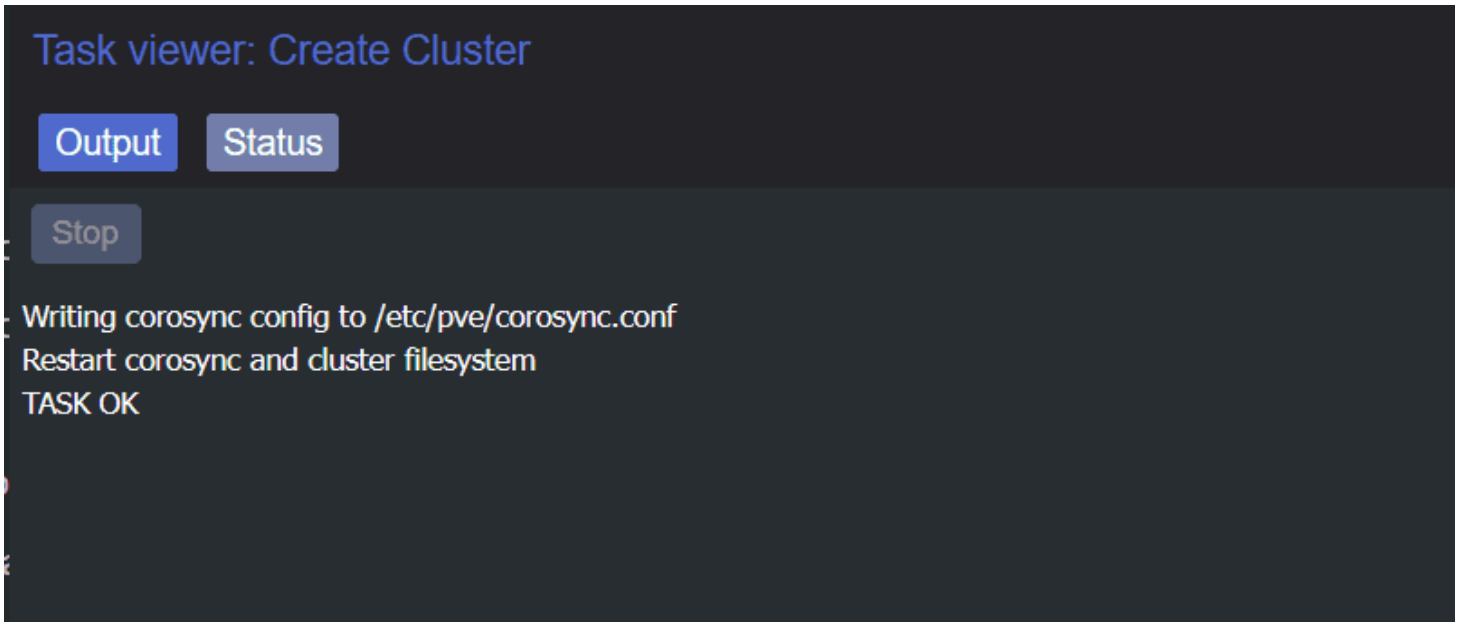


You will use this join information to join cluster on the second, and third node.

Step 4: Configure the corosync communication protocol

To configure the corosync communication protocol, click on the “Cluster” tab in the Proxmox Web GUI and then click on the “Edit” button next to the cluster you want to configure.

This will open a dialog where you can modify the settings for the corosync communication protocol, including the communication port and the number of votes required to reach quorum.



Step 5: Add virtual machines to the cluster

Once the cluster has been configured, you can add virtual machines to the cluster. To do this, click on the “Virtual Machines” tab in the Proxmox Web GUI and then click on the “Create VM” button.

This will open a dialog where you can create and configure virtual machines, including specifying the virtual machine name, the operating system, and the storage location.

Step 6: Monitor the cluster

To ensure the reliability of your virtual environment, it is important to monitor the cluster and to perform regular maintenance tasks. This can be done using the Proxmox Web GUI by clicking on the “Cluster” tab and then clicking on the “Monitor” button.

This will provide information on the status of the nodes in the cluster and will allow you to perform tasks such as live migrations of virtual machines

Cluster cold start

Cluster cold start refers to the process of starting a Proxmox Cluster from scratch, without any previous configuration or state information. A cluster cold start is typically performed in the following scenarios:

1. After a complete failure of the cluster: In the event of a complete failure of the cluster, all configuration information and state information are lost, and a cluster cold start is necessary to rebuild the cluster from scratch.

2. When setting up a new Proxmox Cluster: When setting up a new Proxmox Cluster, a cluster cold start is necessary to create a new cluster and configure the cluster from scratch.

3. When changing the cluster configuration: When changing the configuration of an existing Proxmox Cluster, such as adding or removing nodes, a cluster cold start may be necessary to properly reconfigure the cluster.

A cluster cold start in Proxmox Clusters involves installing Proxmox on each node, configuring the network settings, creating a new cluster, adding nodes to the cluster, and configuring the corosync communication protocol. This process

can be performed using the Proxmox Web GUI or by using the command line.

It is important to note that a cluster cold start can result in data loss, as all virtual machines and configurations will need to be recreated. As such, it is important to plan properly and back up all virtual machines and configurations prior to performing a cluster cold start.

Wrapping Up

Proxmox is a great platform for running home lab workloads and production environments. With Proxmox clusters, you can set up a high-availability environment to protect your virtual machines from a single node failure in the data center.

If you follow all the steps listed to create a Proxmox cluster, you can easily create a Proxmox cluster using the web UI and CLI.

Mastering Ceph Storage Configuration in Proxmox 8 Cluster

June 26, 2023
Proxmox

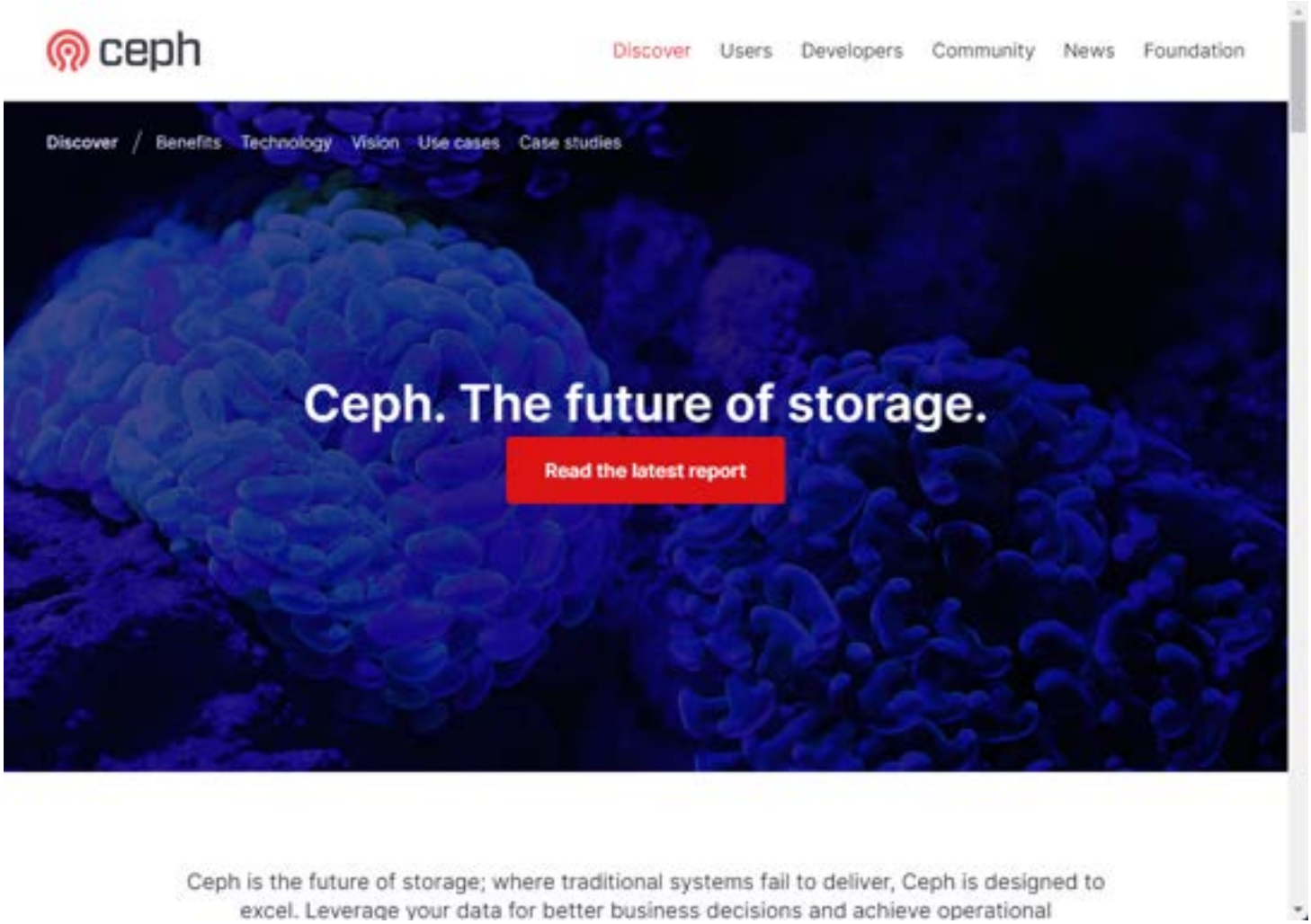
Mastering Ceph Storage Configuration in Proxmox 8 Cluster

The need for highly scalable storage solutions that are fault-tolerant and offer a unified system is undeniably significant in data storage. One such solution is Ceph Storage, a powerful and flexible storage system that facilitates data replication and provides data redundancy. In conjunction with Proxmox, an open-source virtualization management platform, it can help manage important business data with great efficiency. Ceph Storage is an excellent storage platform because it's designed to run on commodity hardware, providing an enterprise-level deployment experience that's both cost-effective and highly reliable. Let's look at mastering Ceph Storage configuration in Proxmox 8 Cluster.

Table of contents

What is Ceph Storage?

Ceph Storage is an open-source, highly scalable storage solution designed to accommodate object storage devices, block devices, and file storage within the same cluster. In essence, Ceph is a unified system that aims to simplify the storage and management of large data volumes.



Ceph Storage is an open source and robust storage solution

A Ceph storage cluster consists of several different types of daemons: Ceph OSD Daemons (OSD stands for Object Storage Daemon), Ceph Monitors, Ceph MDS (Metadata Server or metadata server cluster), and others. Each daemon type plays a distinct role in the operation of the storage system.

Ceph OSD Daemons handle data storage and replication, storing the data across different devices in the cluster. The Ceph Monitors, on the other hand, track the cluster state, maintaining a map of the entire system, including all the data and daemons.

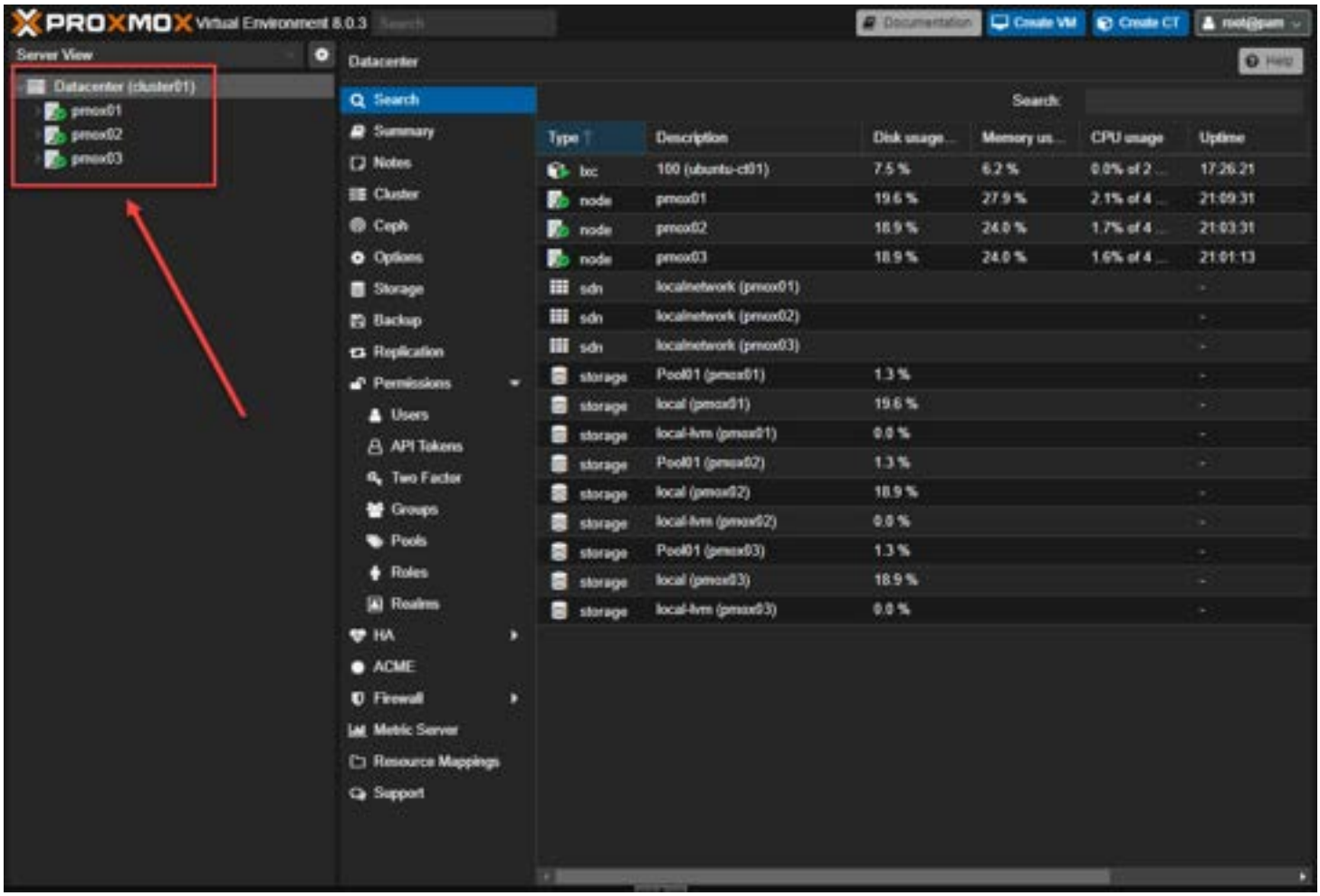
Ceph MDS, or metadata servers, are specific to the Ceph File System. They store metadata for the filesystem, which allows the Ceph OSD Daemons to concentrate solely on data management.

A key characteristic of Ceph storage is its intelligent data placement method. An algorithm called CRUSH (Controlled Replication Under Scalable Hashing) decides where to store and how to retrieve data, avoiding any single point of failure and effectively providing fault-tolerant storage.

What is a Proxmox Cluster and Why is Shared Storage Needed?

A Proxmox cluster is a group of Proxmox VE servers working together. These servers, known as nodes, share resources and operate as a single system. Clustering allows for central management of these servers, making it easier to manage resources and distribute workloads across multiple nodes.

Below, I have created a new Proxmox 8 cluster of three nodes.



Proxmox 8 cluster

Shared storage systems

Shared storage is essential in a Proxmox cluster for several reasons. Firstly, it enables high availability. If one node fails, the virtual machines (VMs) or containers running on that node can be migrated to another node with minimal downtime. Shared storage ensures the data those VMs or containers need is readily available on all nodes.

Secondly, shared storage facilitates load balancing. You can easily move VMs or containers from one node to another, distributing the workload evenly across the cluster. This movement enhances performance, as no single node becomes a bottleneck.

Lastly, shared storage makes data backup and recovery more manageable. With all data centrally stored, it's easier to implement backup strategies and recover data in case of a failure. In this context, Ceph, with its robust data replication and fault tolerance capabilities, becomes an excellent choice for shared storage in a Proxmox cluster.

Why is Ceph Storage a Great Option in Proxmox for Shared Storage?

Proxmox supports several block and object storage solutions. Ceph Storage brings Proxmox a combination of scalability, resilience, and performance that few other storage systems can offer. With its unique ability to simultaneously offer object, block, and file storage, Ceph can meet diverse data needs, making it an excellent choice for shared storage in a Proxmox environment.

One of the key reasons that Ceph is a great option for shared storage in Proxmox is its scalability. As your data grows, Ceph can effortlessly scale out to accommodate the increased data volume. You can add more storage nodes to your cluster at any time, and Ceph will automatically start using them.

Fault tolerance is another reason why Ceph is a great choice. With its inherent data replication and redundancy, you can lose several nodes in your cluster, and your data will still remain accessible and intact. In addition to this, Ceph is designed to recover automatically from failures, meaning that it will strive to replicate data to other nodes if one fails.

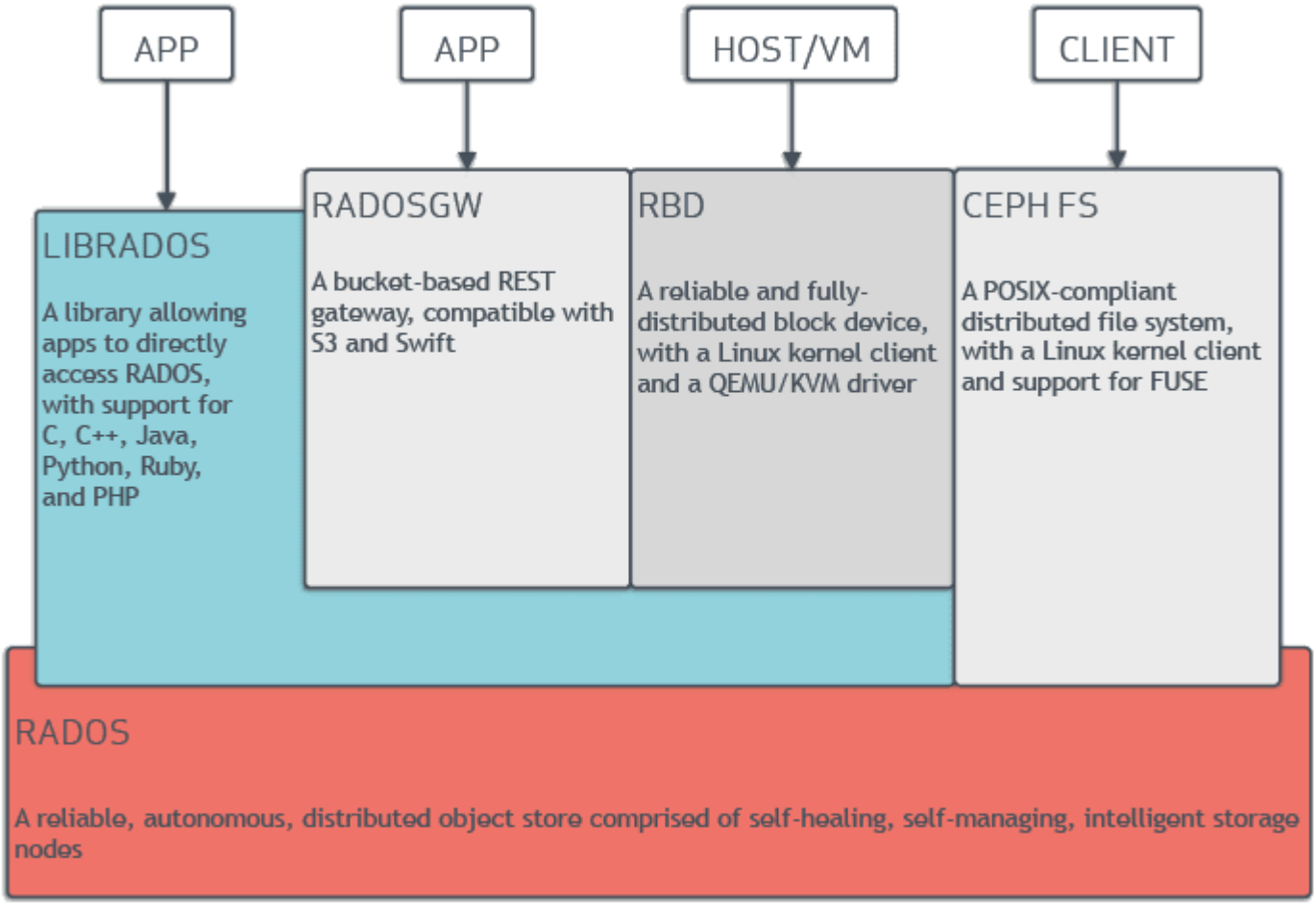
Ceph's integration with Proxmox for shared storage enables virtual machines and containers in the Proxmox environment to leverage the robust Ceph storage system. This integration makes Ceph an even more attractive solution, as it brings its strengths into a virtualized environment, further enhancing Proxmox's capabilities.

Finally, Ceph's ability to provide object, block, and file storage simultaneously allows it to handle a wide variety of workloads. This versatility means that whatever your shared storage needs, Ceph in a Proxmox environment is likely to be a solution that can handle it effectively and efficiently.

Understanding the Ceph Storage Cluster

At its core, a Ceph storage cluster consists of several components, each having a specific role in the storage system. These include Ceph OSDs (Object Storage Daemons), which manage data storage, and Ceph Monitors, which maintain the cluster state. The CRUSH algorithm controls data placement, enabling scalable hashing and avoiding any single point of failure in the cluster. Ceph MDS (Metadata Servers) are also part of this structure, which store metadata associated with the Ceph filesystem.

Ceph clients interface with these components to read and write data, providing a robust, fault-tolerant solution for enterprise-level deployments. The data stored in the cluster is automatically replicated to prevent loss, thanks to controlled replication mechanisms.



Ceph Storage Architecture

Configuring the Proxmox and Ceph Integration

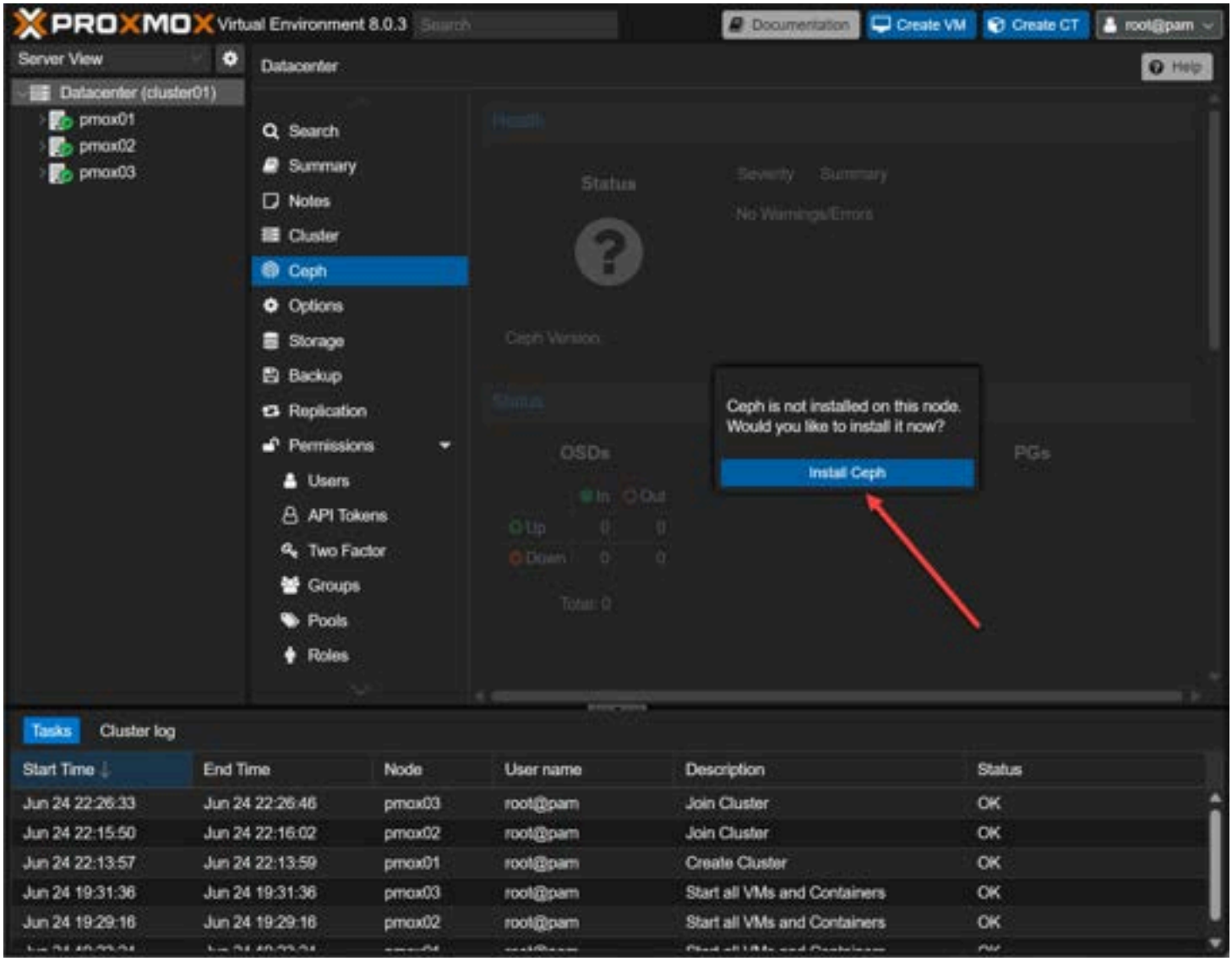
Proxmox offers a user-friendly interface for integrating Ceph storage clusters into your existing infrastructure. This integration harnesses the combined power of object, block, and file storage, offering a versatile data storage solution.

Before you begin, ensure that your Proxmox cluster is up and running, and the necessary Ceph packages are installed. It's essential to note that the configuration process varies depending on the specifics of your existing infrastructure.

Installing and Configuring Ceph

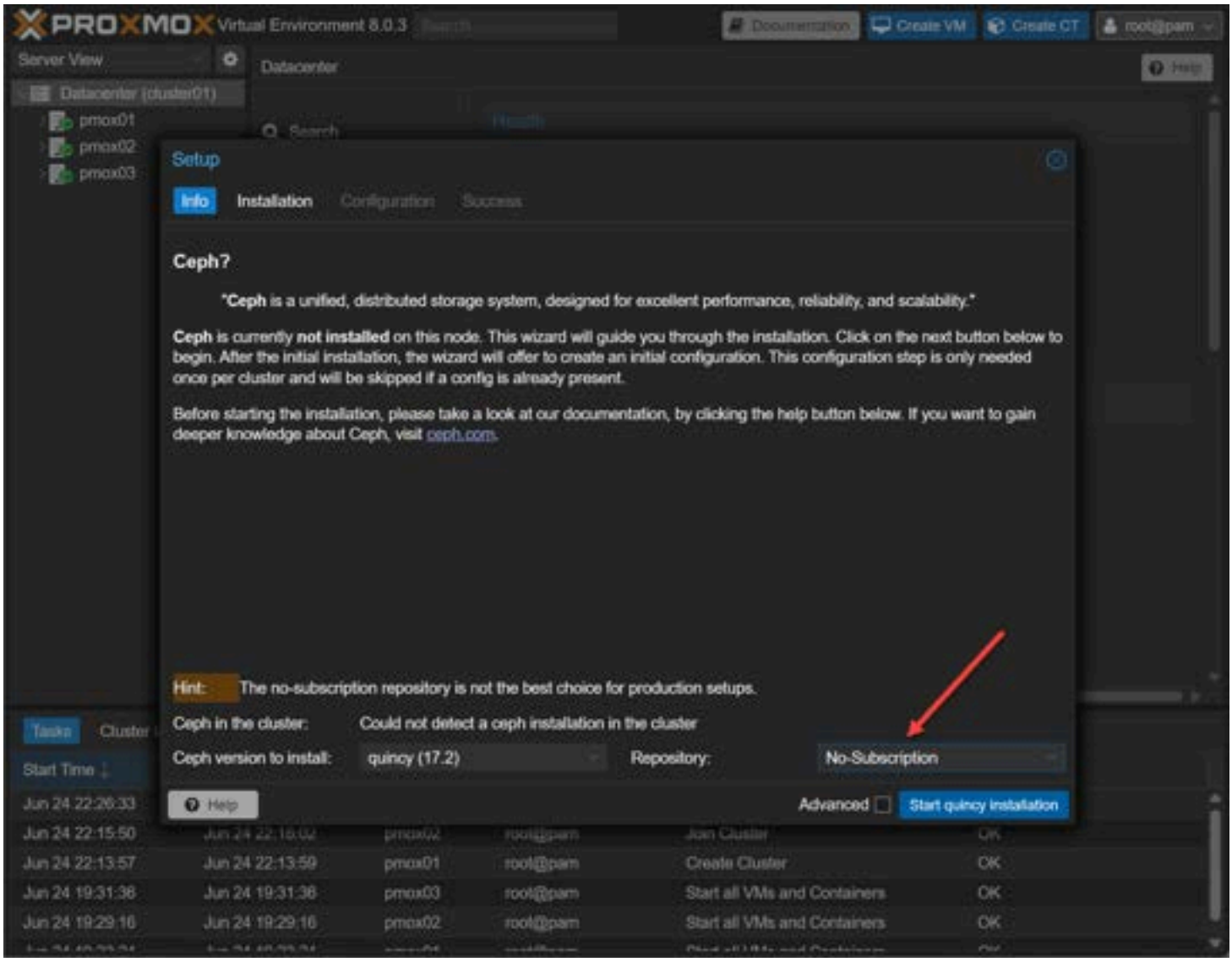
Start by installing the Ceph packages in your Proxmox environment. These packages include essential Ceph components like Ceph OSD daemons, Ceph Monitors (Ceph Mon), and Ceph Managers (Ceph Mgr).

Click on one of your Proxmox nodes, and navigate to **Ceph**. When you click Ceph, it will prompt you to install Ceph.



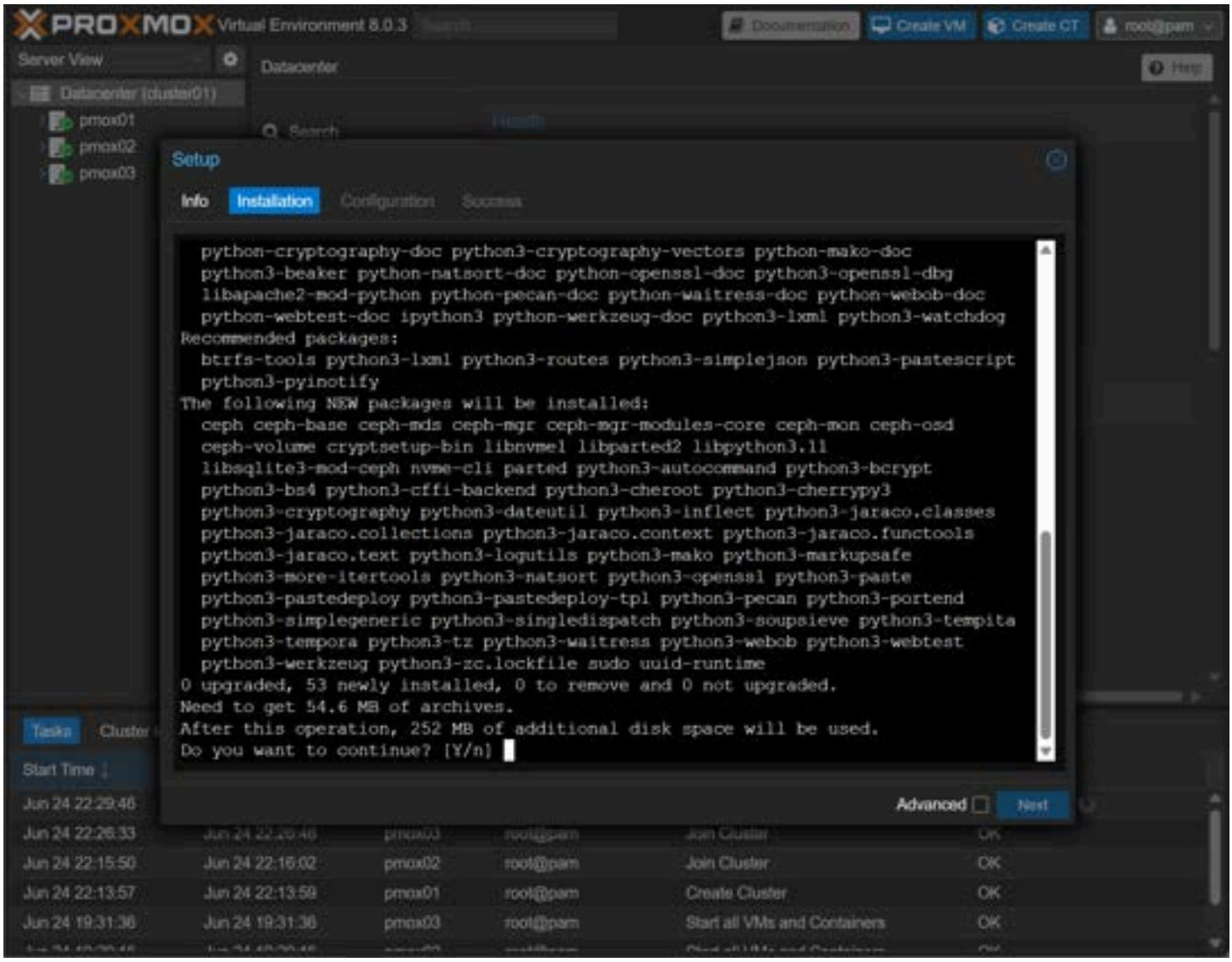
Install Ceph on each Proxmox 8 cluster node

This begins the setup wizard. First, you will want to choose your **Repository**. This is especially important if you don't have a subscription. You will want to choose the **No Subscription** option. For production environments, you will want to use the **Enterprise** repository.

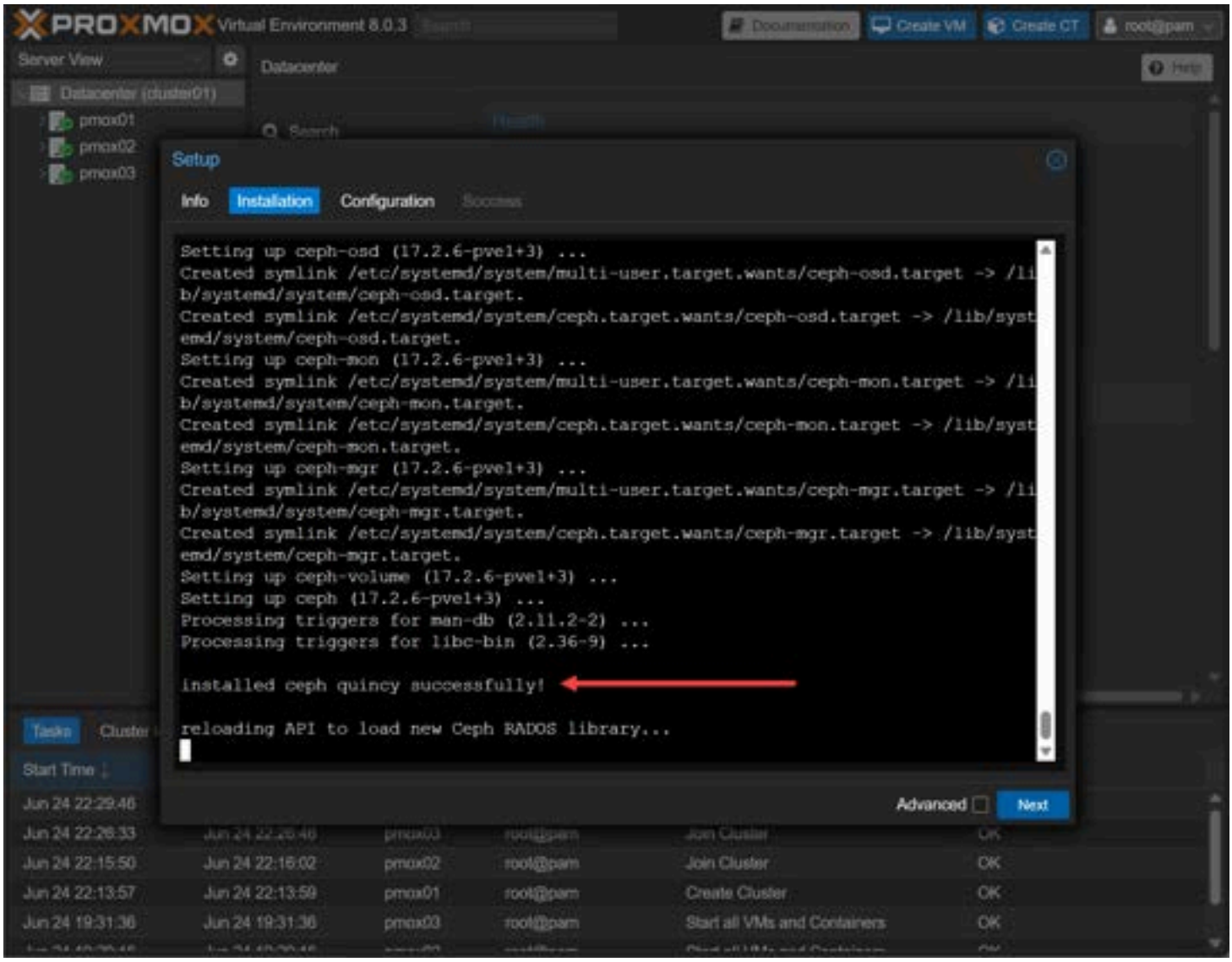


Choosing the Ceph repository and beginning the installation

You will be asked if you want to continue the installation of Ceph. Type **Y** to continue.

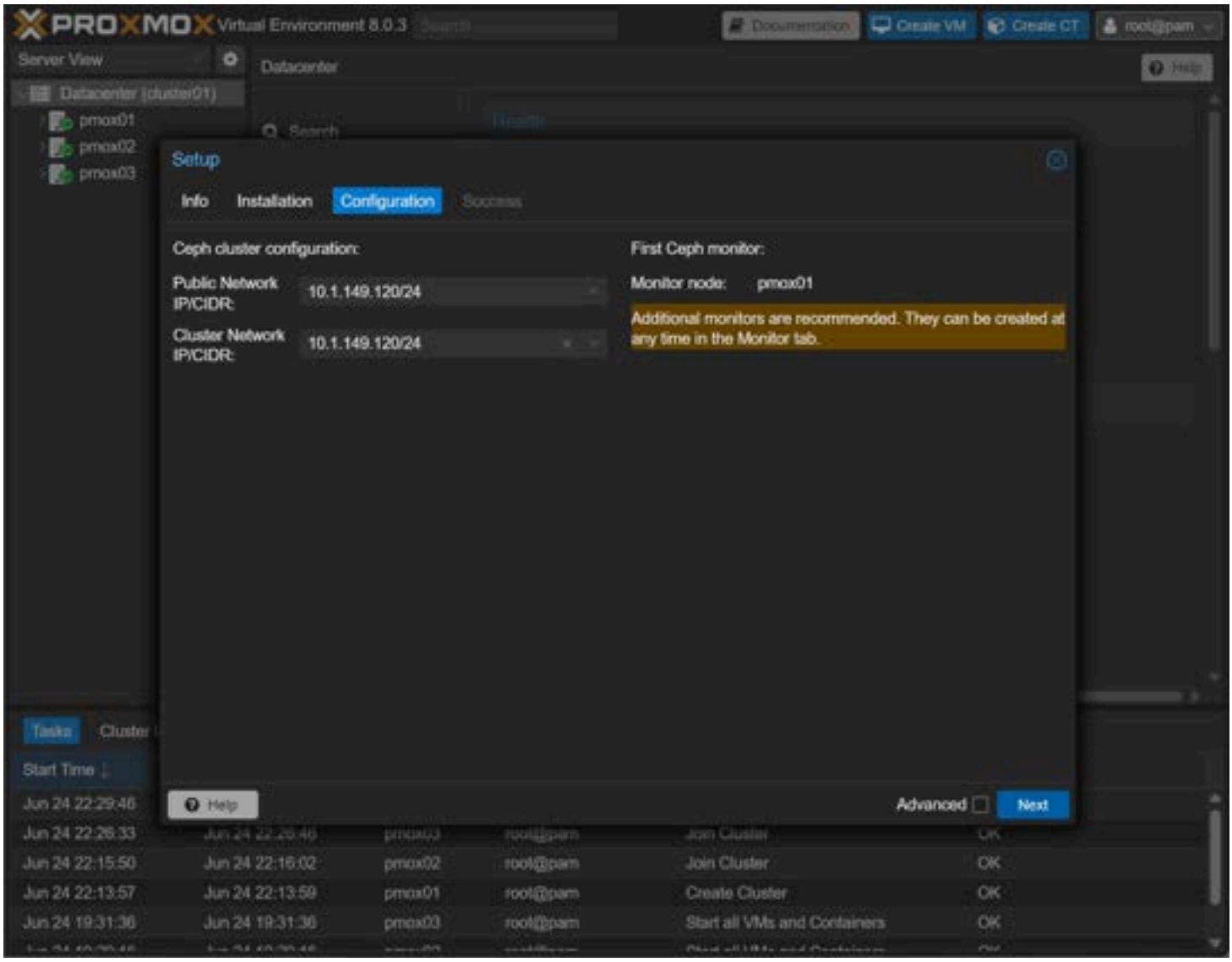


Verify the installation of Ceph storage modules



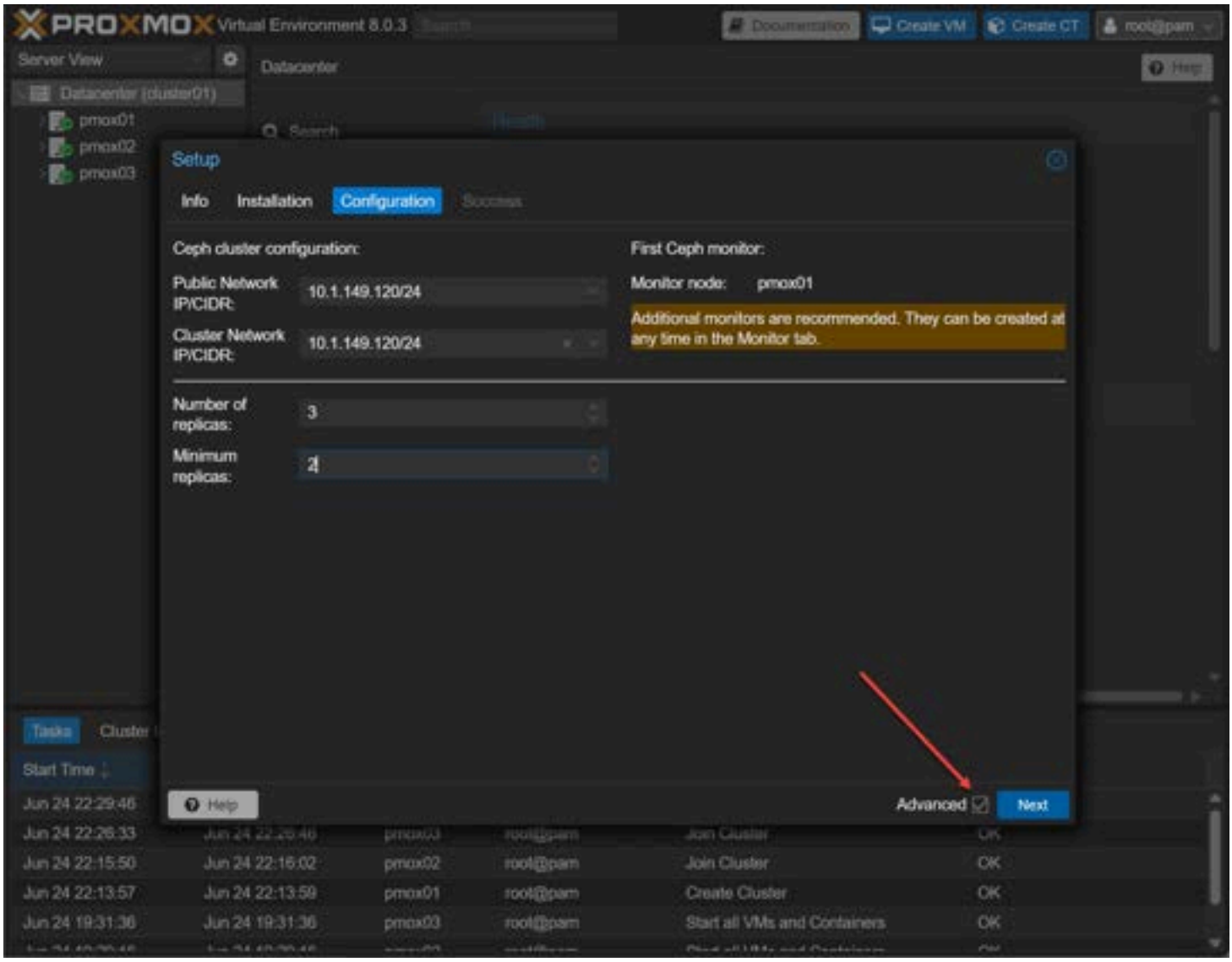
Ceph installed successfully

Next, you will need to choose the **Public Network** and the **Cluster Network**. Here, I don't have dedicated networks configured since this is a nested installation. So I am just choose the same subnet for each.



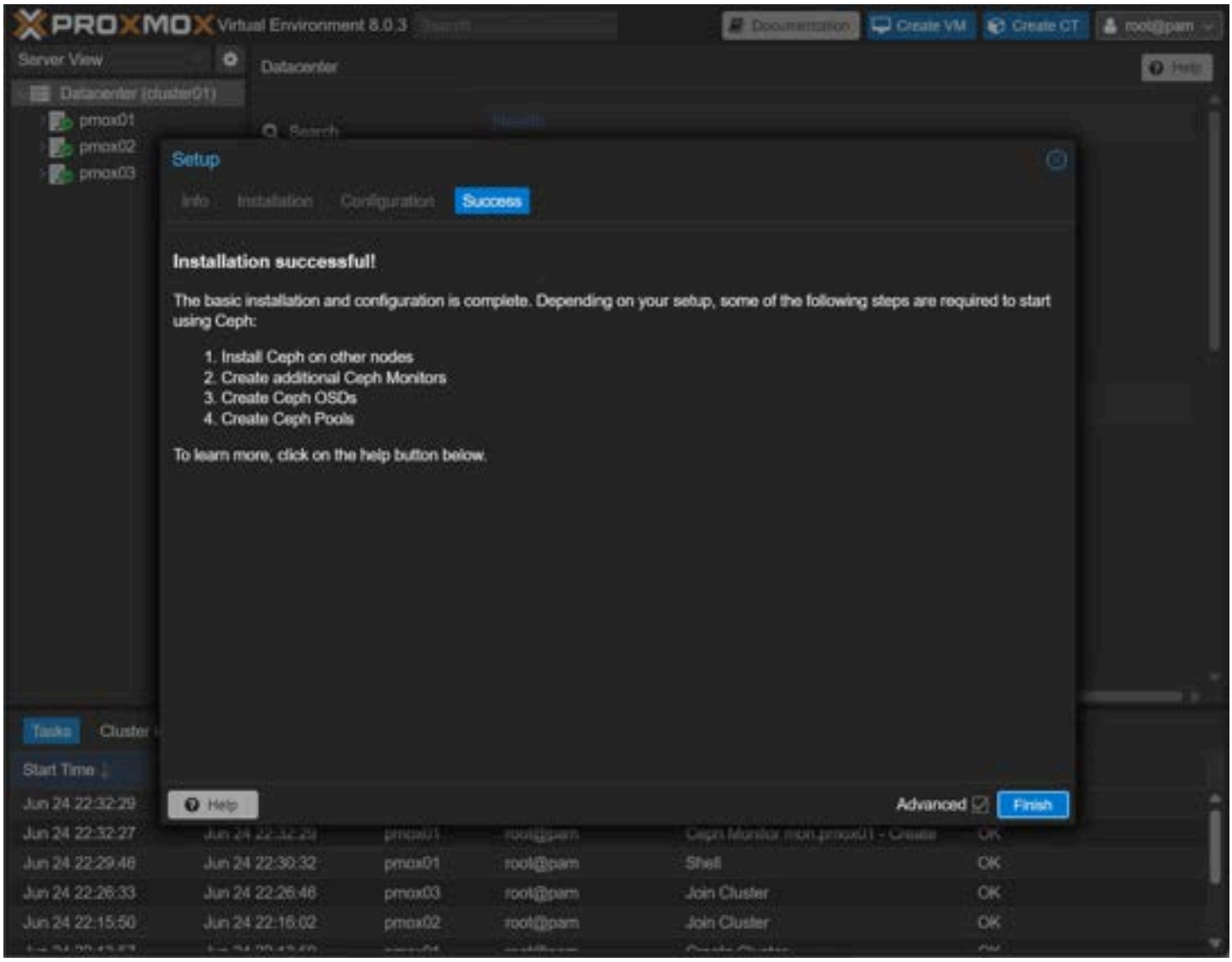
Configuring the public and cluster networks

If you click the **Advanced** checkbox, you will be able to setup the **Number of replicas** and **Minimum replicas**.



Advanced configuration including the number of replicas

At this point, Ceph has been successfully installed on the Proxmox node.



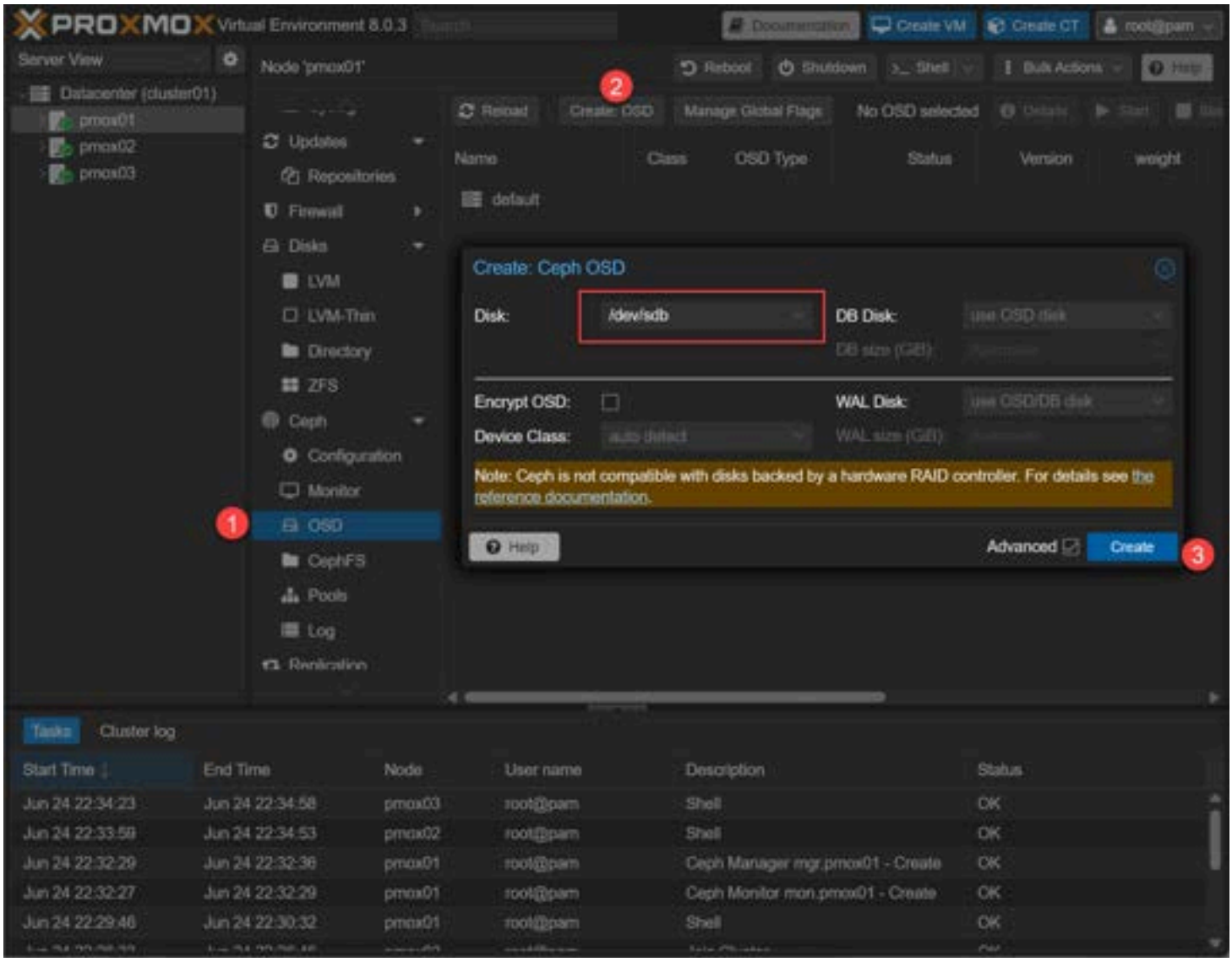
Ceph configured successfully and additional setup steps needed

Repeat these steps on the remaining cluster nodes in your Proxmox cluster configuration.

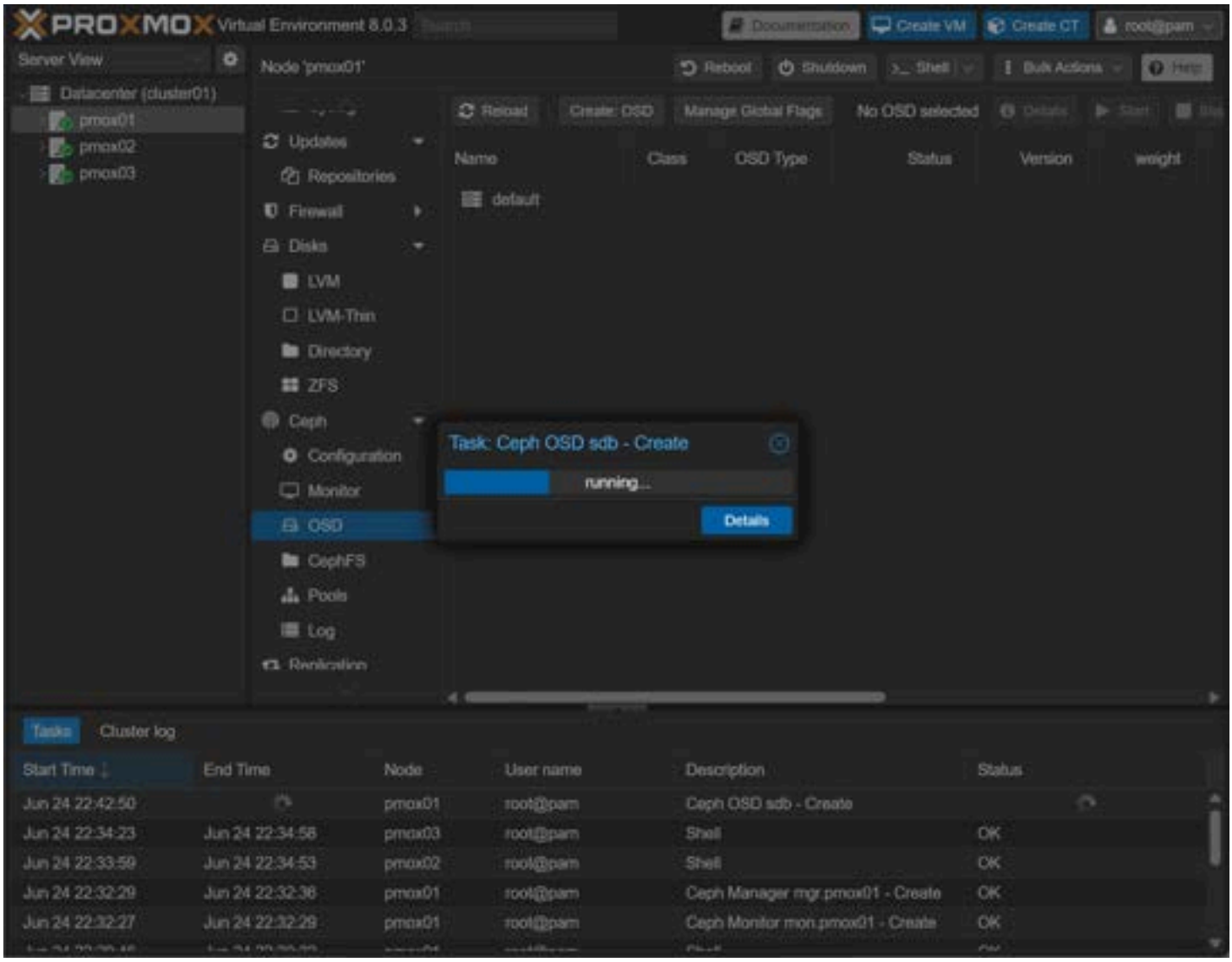
Setting up Ceph OSD Daemons and Ceph Monitors

Ceph OSD Daemons and Ceph Monitors are crucial to the operation of your Ceph storage cluster. The OSD daemons handle data storage, retrieval, and replication on the storage devices, while Ceph Monitors maintain the cluster map, tracking active and failed cluster nodes.

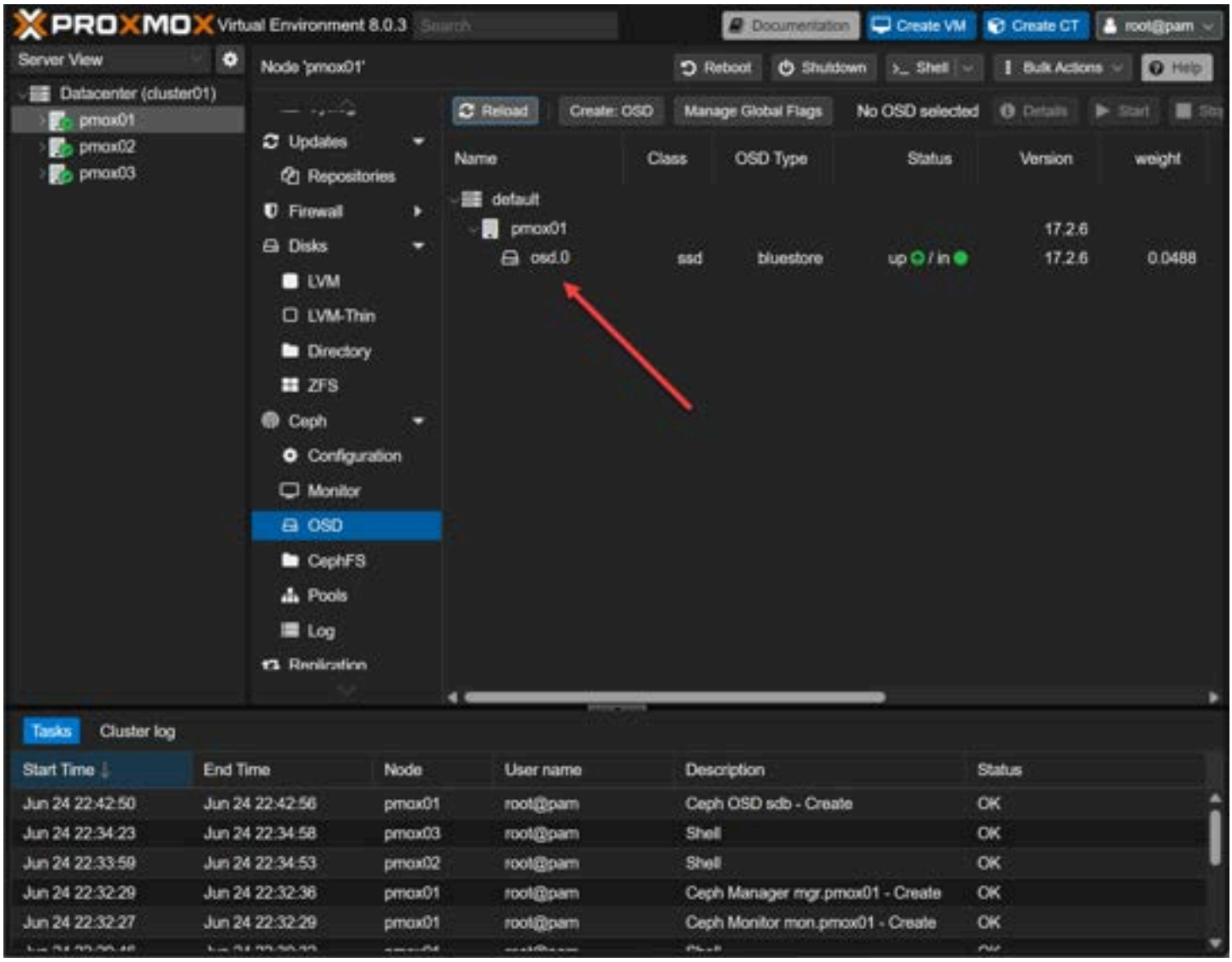
You'll need to assign several Ceph OSDs to handle data storage and maintain the redundancy of your data.



Adding an OSD in Proxmox Ceph storage

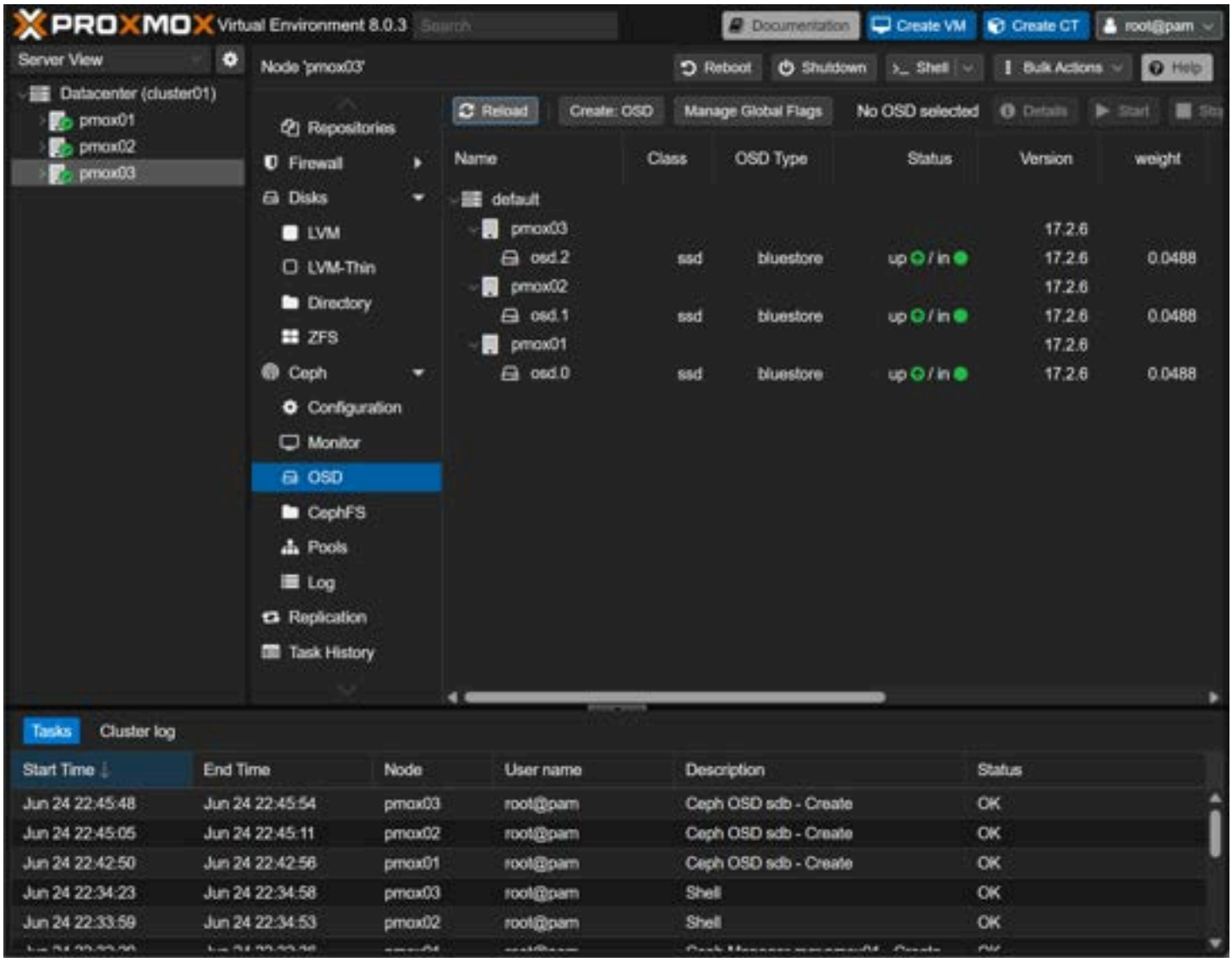


The OSD begins configuring and adding



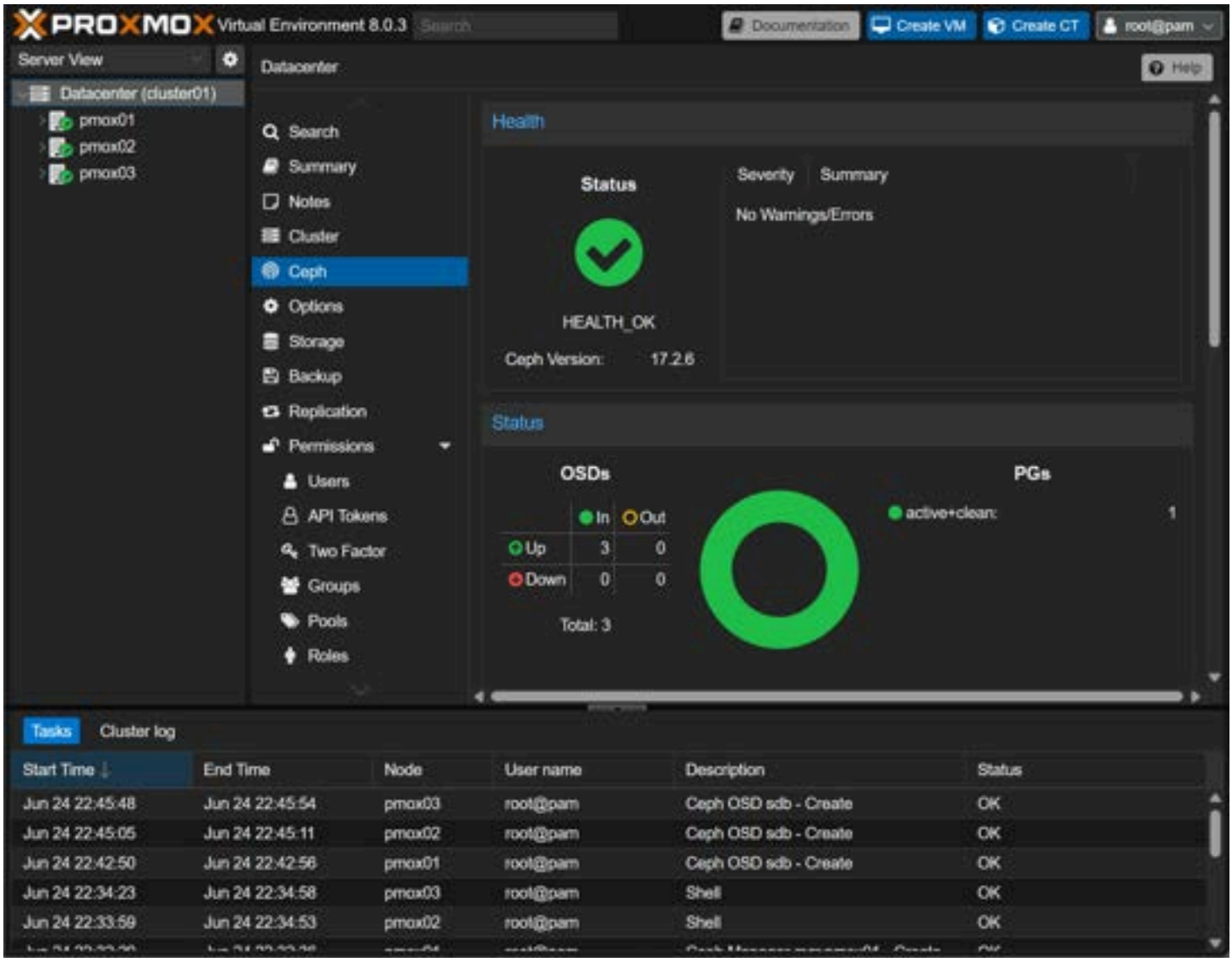
The OSD is successfully added to the Proxmox host

Also, set up more than one Ceph Monitor to ensure high availability and fault tolerance.



OSDs added to all three Proxmox nodes

At this point, if we visit the Ceph storage dashboard , we will see the status of the Ceph storage cluster.



Healthy Ceph storage status for the cluster

Creating Ceph Monitors

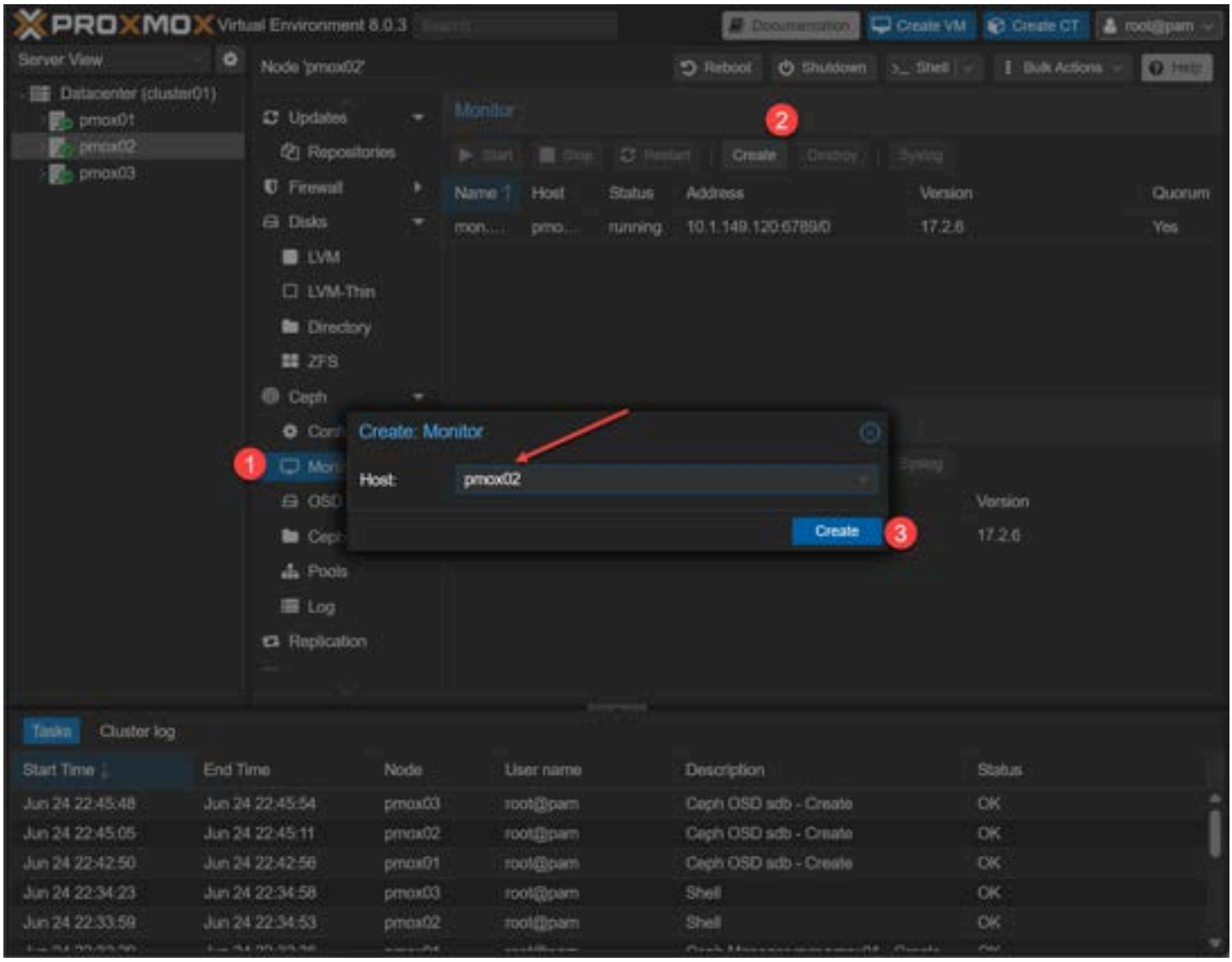
Let's add additional Ceph Monitors, as we have only configured the first node as a Ceph monitor. Whatis a Ceph Monitor?

A Ceph Monitor, often abbreviated as Ceph Mon, is an essential component in a Ceph storage cluster. Its primary function is to maintain and manage the cluster map, a crucial data structure that keeps track of the entire cluster's state, including the location of data, the cluster topology, and the status of other daemons in the system.

Ceph Monitors contribute significantly to the cluster's fault tolerance and reliability. They work in a quorum, meaning there are multiple monitors, and a majority must agree on the cluster's state. This setup prevents any single point of failure, as even if one monitor goes down, the cluster can continue functioning with the remaining monitors.

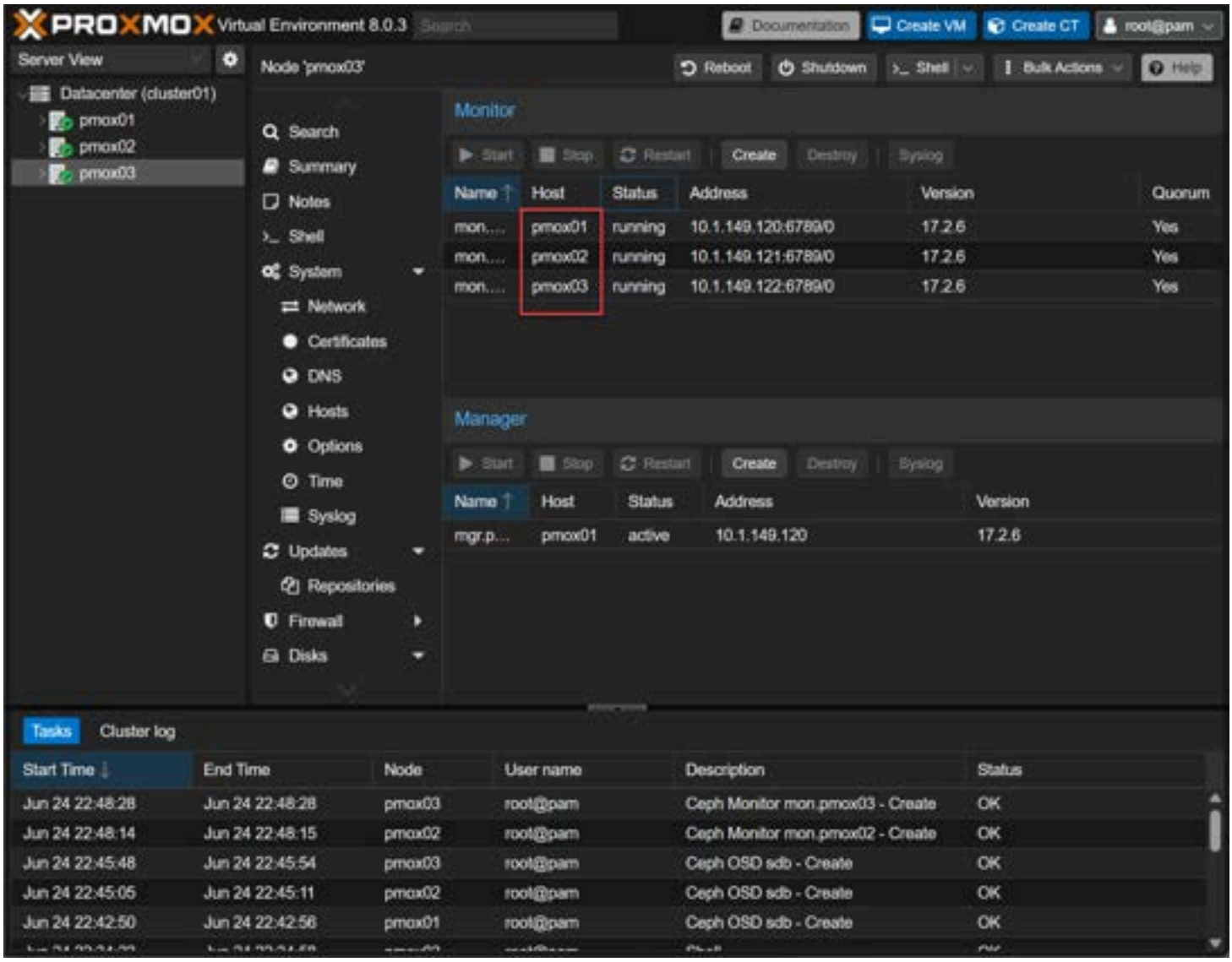
By keeping track of the data locations and daemon statuses, Ceph Monitors facilitate efficient data access and help ensure the seamless operation of the cluster. They are also involved in maintaining data consistency across the cluster and managing client authentication and authorization.

Here we are adding the 2nd Proxmox node as a monitor. I added the 3rd one as well.



Adding Ceph Monitors to additional Proxmox hosts

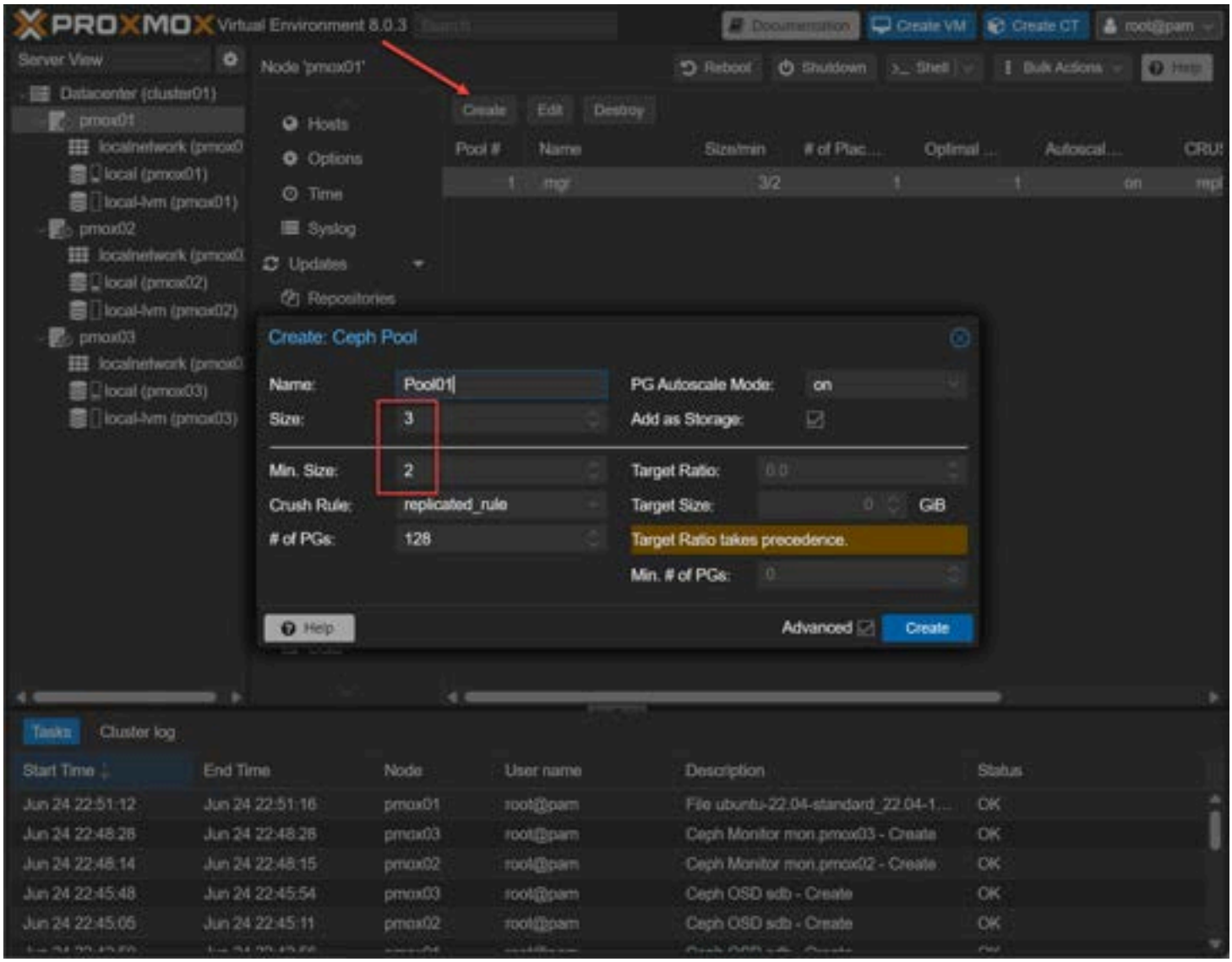
Now, each node is a monitor.



All three Proxmox hosts are running the Ceph Monitor

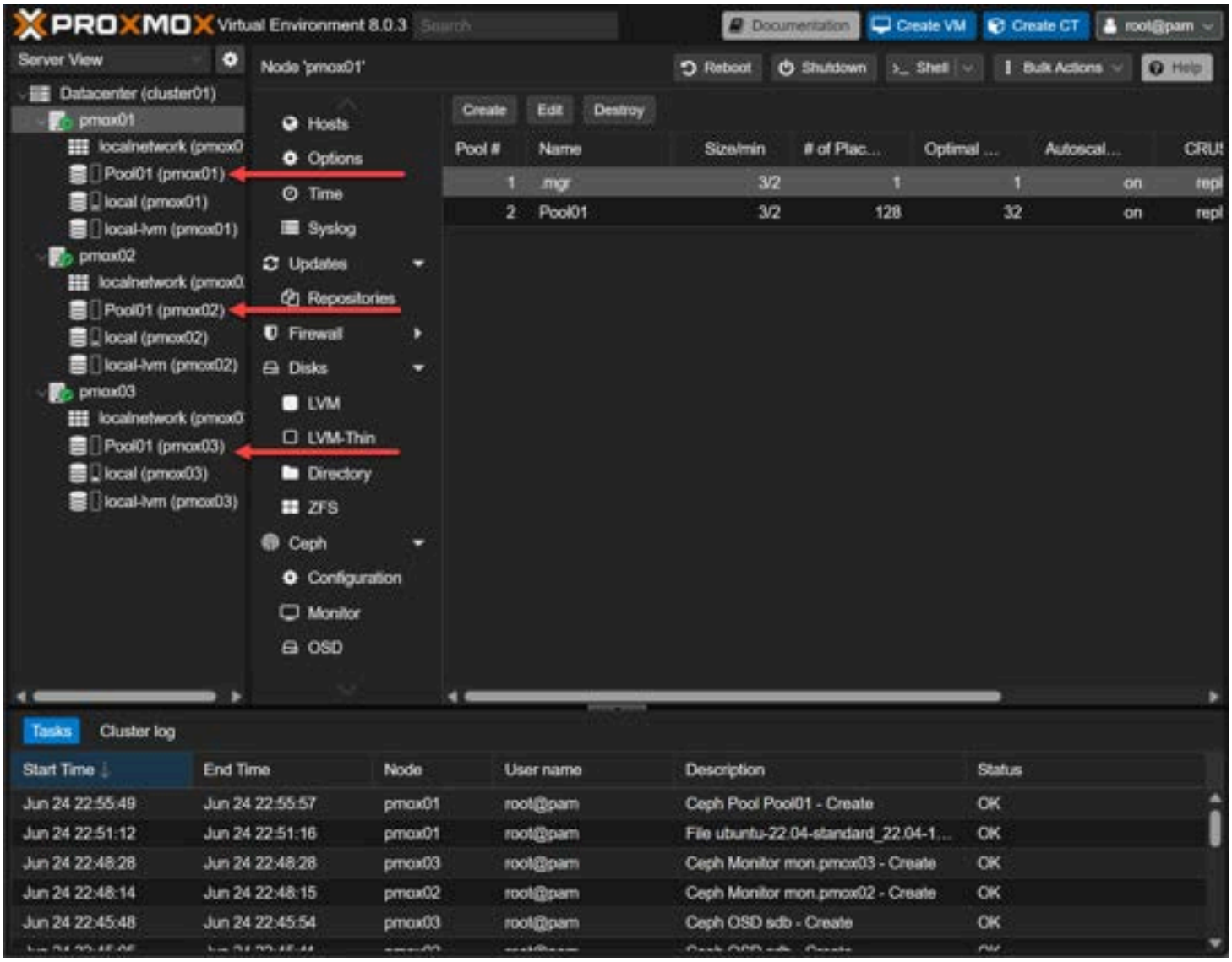
Creating a Ceph Pool for VM and Container storage

Now that we have the OSDs and Monitors configured, we can create our Ceph Pool. Below we can see the replicas and minimum replicas.



Creating a Ceph Pool

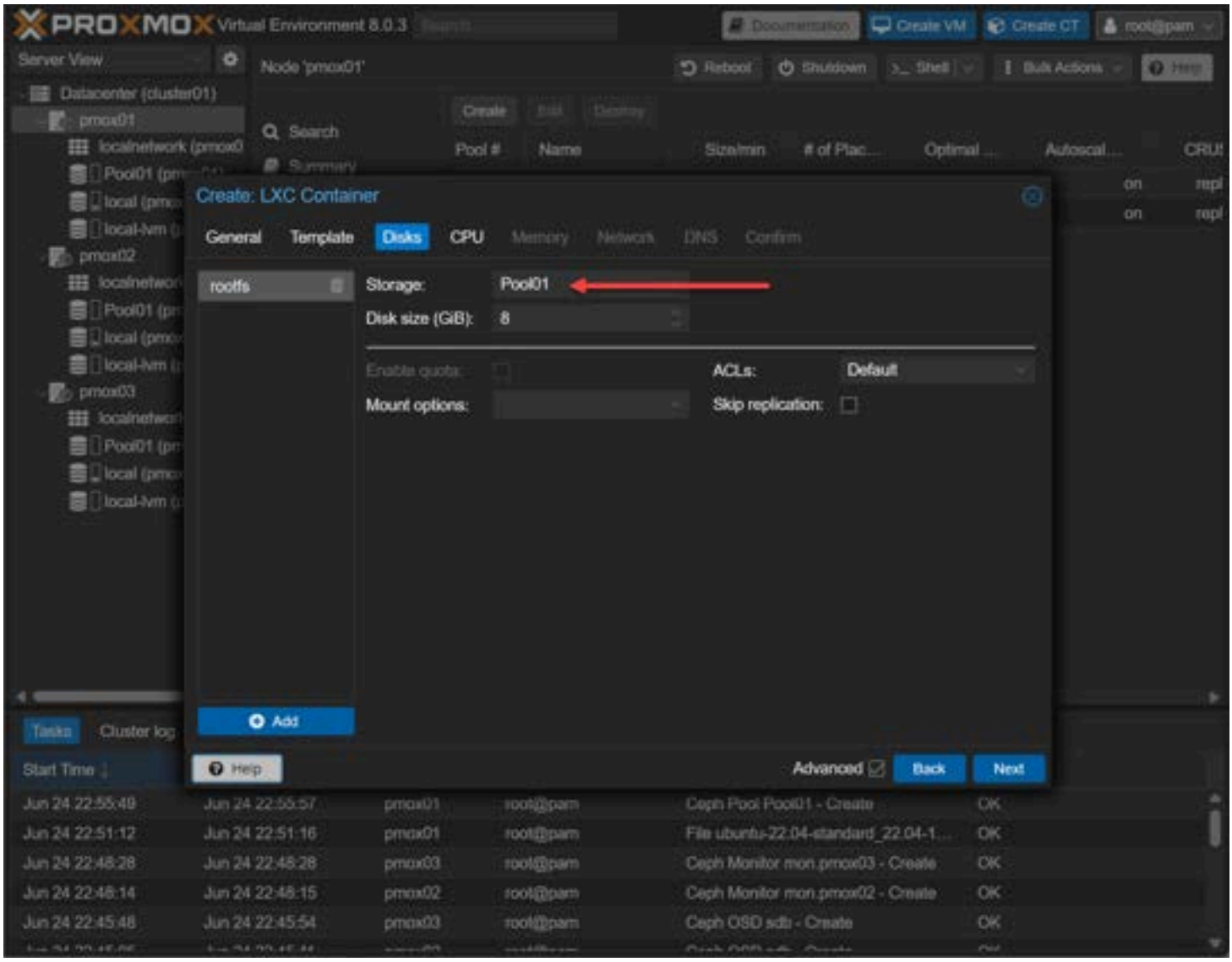
Now, the Ceph Pool is automatically added to the Prommox cluster nodes.



Pool added to all three Proxmox nodes

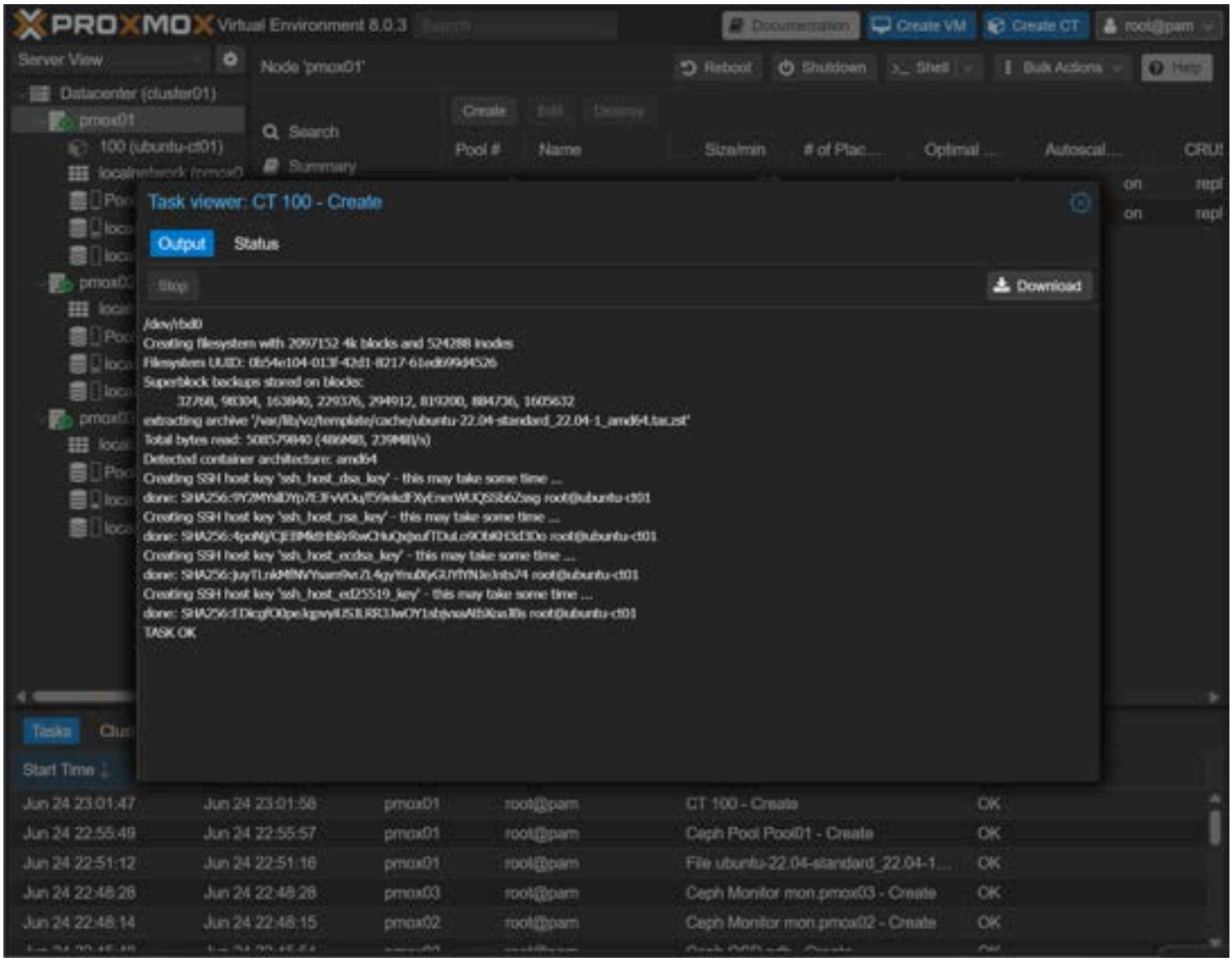
Utilizing Ceph storage for Virtual Machines and Containers

Now that we have the Ceph Pool configured, we can use it for backing storage for Proxmox Virtual Machines and Containers. Below, I am creating a new LXC container. Note how we can choose the new Ceph Pool as the container storage.



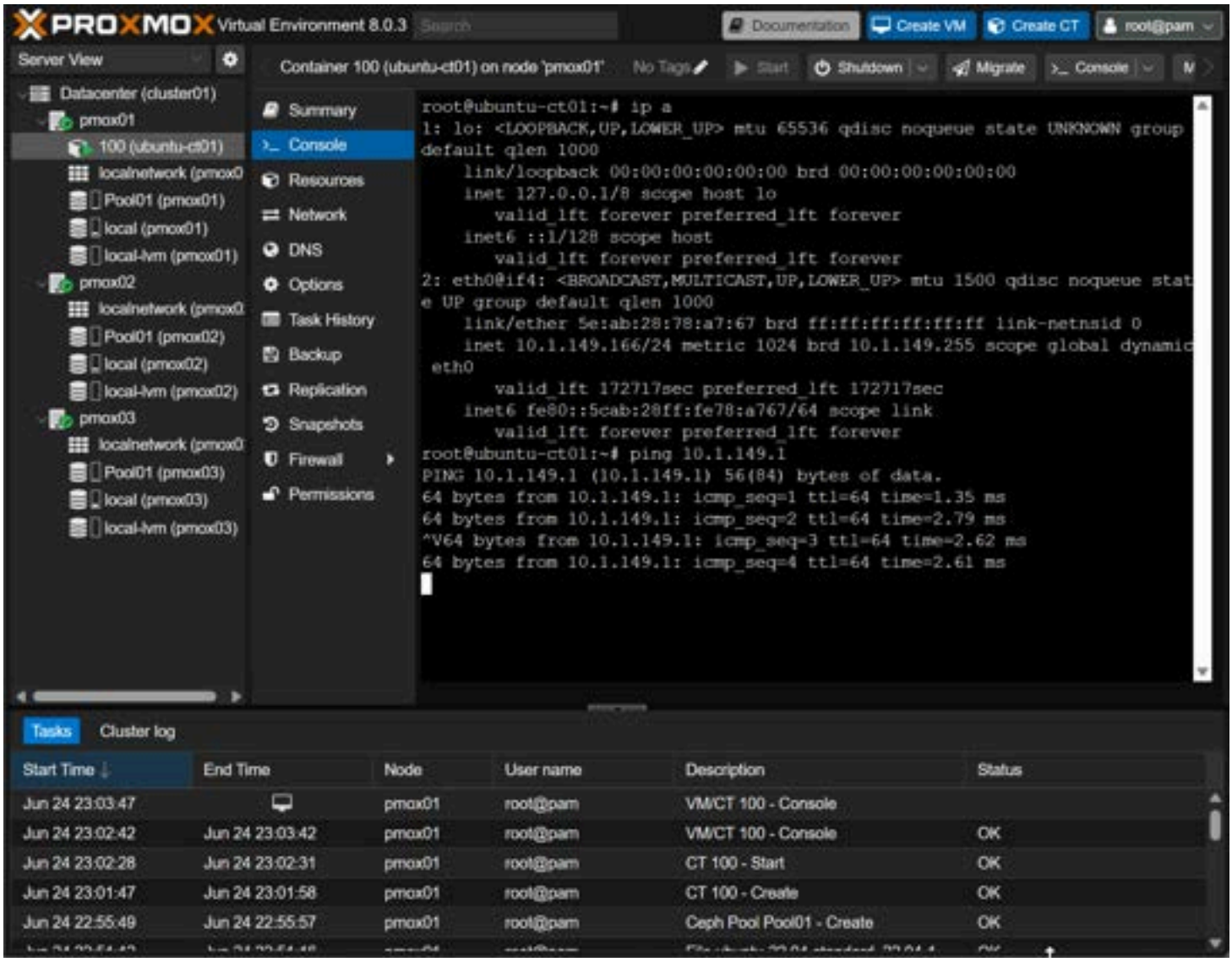
Choosing the new pool for Proxmox LXC container storage

The LXC container creates successfully with no storage issues which is good.

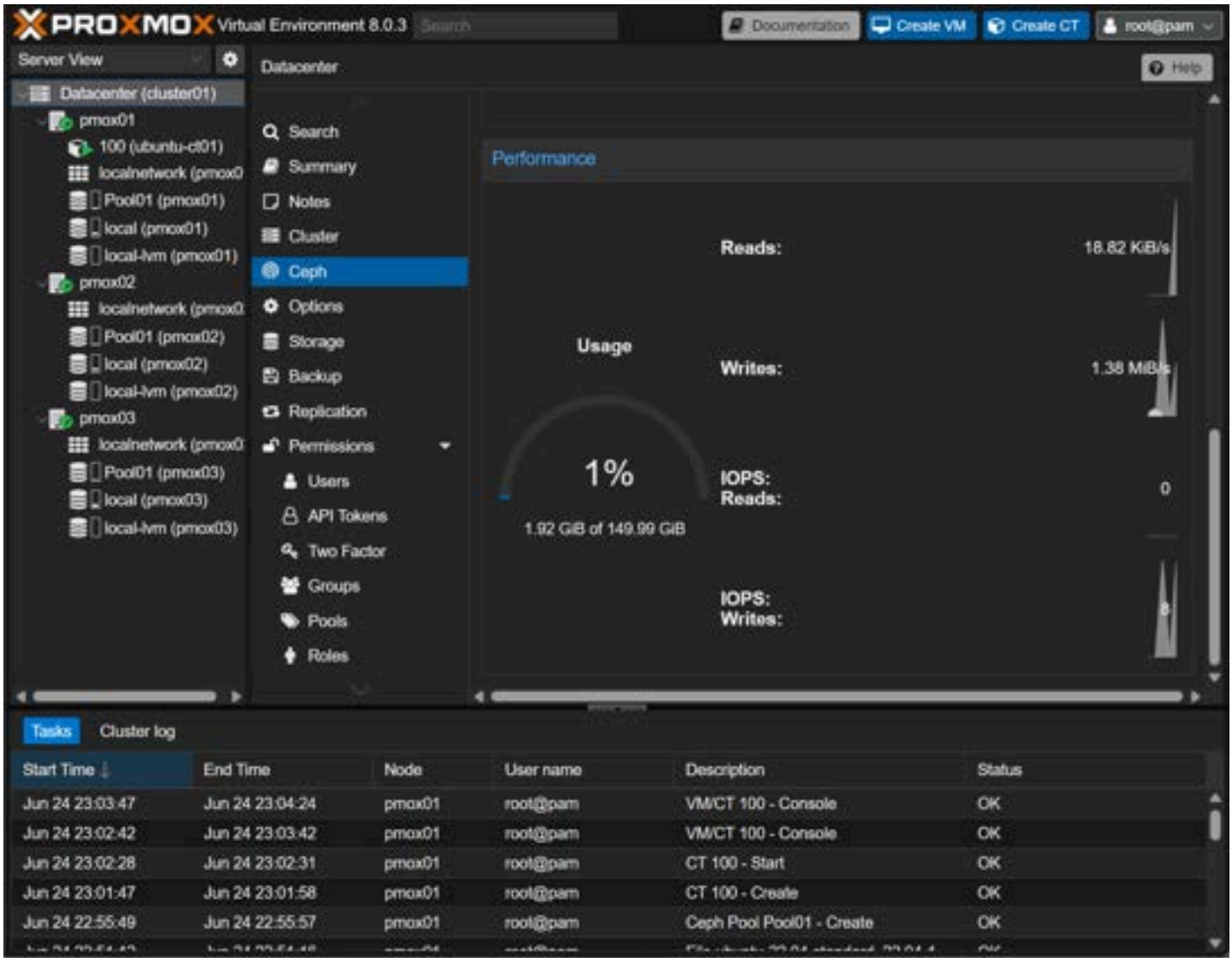


The new LXC container is created successfully on Ceph storage

We can see we have the container up and running without issue. Also, I was able to migrate the LXC container to another node without issue.



The LXC container operating on the Ceph Pool



Ceph performance dashboard in Proxmox

Now, let's learn a little more about Ceph.

Managing Data with Ceph

Ceph uniquely stores data. It breaks down data into objects before storing them across the cluster. This breakdown of data facilitates scalable storage across multiple storage nodes. It also provides an opportunity to implement erasure coding or redundancy for data protection.

Ceph Object Storage

Object storage in Ceph is done through **RADOS (Reliable Autonomic Distributed Object Store)**. Objects stored are automatically replicated across different storage devices to ensure data availability and fault tolerance. The CRUSH algorithm, a scalable hashing technique, controls how the objects are distributed and accessed, thus avoiding any single point of failure.

Block Storage with Ceph

Ceph Block Devices, or RADOS Block Devices (RBD), is a part of the Ceph storage system that allows Ceph to interact with block storage. These block devices can be virtualized, providing a valuable storage solution for virtual machines in the Proxmox environment. Block storage with Ceph offers features like thin provisioning and cache tiering, further enhancing data storage efficiency.

Ceph File System

The Ceph File System (CephFS) is another significant feature of Ceph. It's a POSIX-compliant file system that uses a Ceph Storage Cluster to store data, allowing for the usual file operations while adding scalability, reliability, and performance.

The Ceph MDS (metadata servers) play a crucial role in the operation of CephFS. They manage file metadata, such as file names and directories, allowing the Ceph OSDs to focus on data storage. This separation improves the overall performance of the Ceph storage system.

Ceph Storage Cluster and Proxmox: A Scalable Storage Solution

You leverage a highly scalable storage solution by configuring a Ceph storage cluster in a Proxmox environment. The combined use of object, block, and file storage methods offers versatile data handling suited to various data types and use cases, such as cloud hosting and cloud-based services.

This combination enables managing important business data effectively while maintaining redundancy and fault tolerance. Whether you're dealing with large file data or smaller objects, using Ceph in a Proxmox environment ensures that your data is safely stored and easily retrievable.

Frequently Asked Questions (FAQs)

How does Ceph Storage achieve fault tolerance?

Ceph storage is inherently fault-tolerant due to its use of controlled data replication. Data stored in a Ceph cluster is automatically replicated across multiple OSDs. Ceph can recover data from other nodes if one node fails, ensuring no data loss. The CRUSH algorithm helps to maintain this fault tolerance by dynamically adjusting the data distribution across the cluster in response to node failures.

Can Ceph Storage handle diverse data types?

Absolutely! Ceph's ability to handle object, block, and file storage makes it versatile and flexible. Ceph uses RADOS Block Devices for block storage, the Ceph filesystem for file storage, and RADOS for object storage. This versatile design enables Ceph to manage diverse data types and workloads, making it an excellent fit for varied data needs.

How does cache tiering enhance Ceph's performance?

Cache tiering is a performance optimization technique in Ceph. It uses smaller, faster storage (like SSDs) as a cache for a larger, slower storage tier. Data is accessed frequently and moved to the cache tier for quicker retrieval. This setup significantly improves read/write performance, making Ceph an excellent option for high-performance applications.

How is Ceph storage beneficial for cloud hosting?

Ceph is a highly scalable, resilient, and performance-oriented storage system, making it an excellent choice for cloud hosting. With its fault tolerance, data replication, and block, object, and file storage support, Ceph can effectively handle the vast and diverse data needs of cloud-based services.

What role do metadata servers play in the Ceph file system?

Metadata servers, or Ceph MDS, manage the metadata for the Ceph filesystem. They handle file metadata such as file names, permissions, and directory structures, allowing the Ceph OSDs to concentrate on data management. This separation boosts performance, making the file system operations more efficient.

Is Ceph Storage a good fit for enterprise-level deployments?

Yes, Ceph is suitable for enterprise-level deployments. Its scalability, robustness, and versatility make it an ideal storage system for large businesses. With its features like thin provisioning, cache tiering, and scalable hashing with the CRUSH algorithm, Ceph can handle vast amounts of data and diverse workloads that large enterprises typically require.

Video covering Proxmox and Ceph configuration

Proxmox 8 Cluster with Ceph Storage configuration
https://youtube.com/watch?v=-qk_P9SKYK4

Proxmox 8 Cluster with Ceph storage

Wrapping up

Ceph storage offers a robust and highly scalable storage solution for Proxmox clusters, making it an excellent option for anyone seeking an efficient way to manage extensive amounts of data and have a highly available storage location for workloads in the home lab or in production. By following this guide, you can implement a Ceph storage cluster in your Proxmox environment and leverage the numerous benefits of this powerful and flexible storage system.

Remember, the versatility of Ceph allows for many configurations tailored to meet specific needs. So, explore the various features of Ceph storage and find a solution that perfectly fits your data storage and management needs.

Other links you may like

- Proxmox 8: New Features and Home Lab Upgrade Instructions
- Proxmox vs ESXi – ultimate comparison 2022
- Docker container security best practices unlocked!
- Nested Proxmox VMware installation in ESXi
- Proxmox Create ISO Storage Location – disk space error

CephFS Configuration in Proxmox Step-by-Step

January 8, 2024
Proxmox

Cephfs configuration in proxmox

Since working with Ceph in Proxmox VE lately, one of the cool features that I wanted to try out was Proxmox CephFS, which allows you to work with your Ceph installation directly from your clients. It allows mounting file storage to your clients on top of your Ceph storage pool with some other really cool benefits. Let's look at CephFS configuration in Proxmox and see how you can install and configure it.

Table of contents

What is CephFS (CephFS file system)?

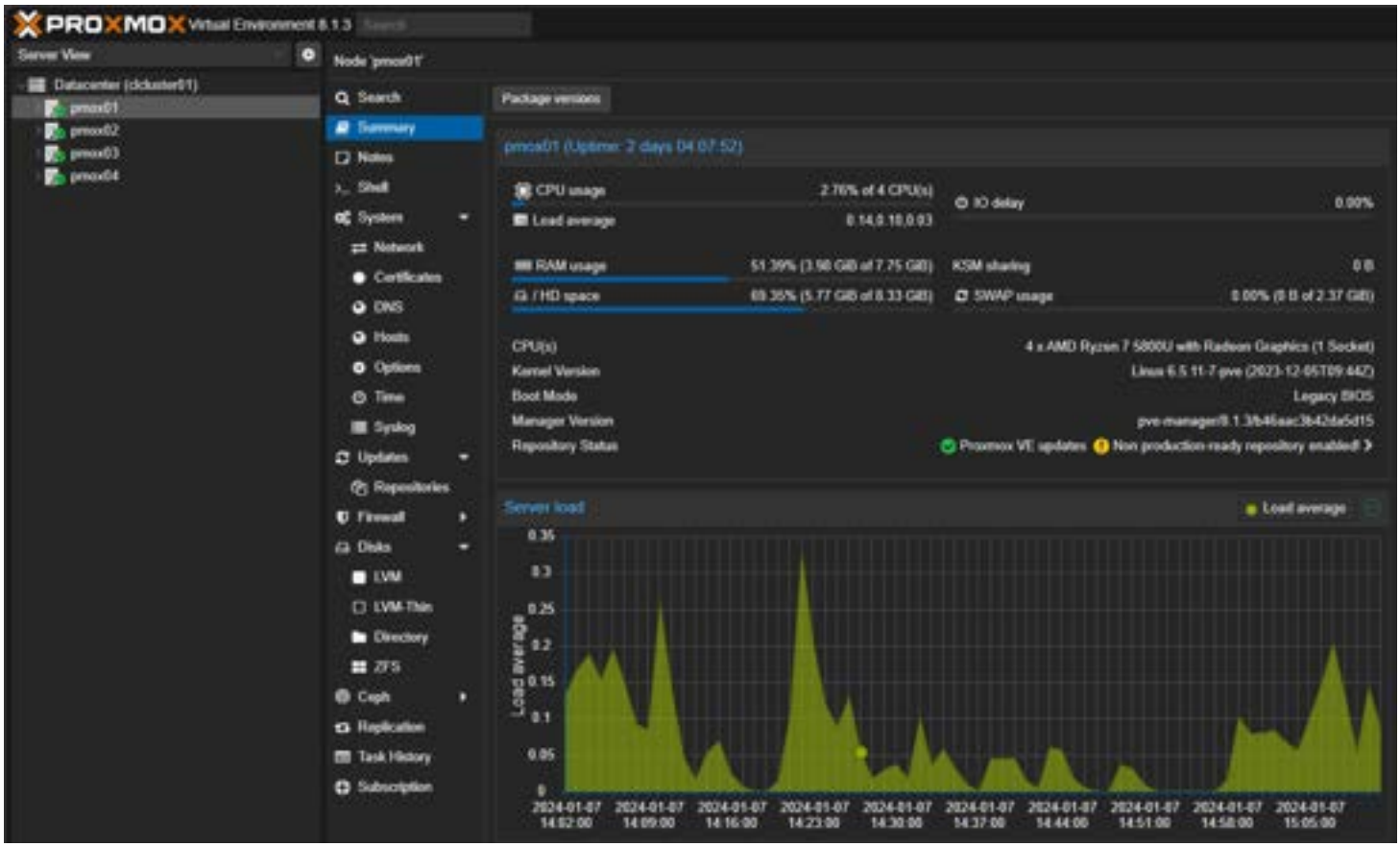
CephFS is a POSIX-compliant file system that offers a scalable and reliable solution for managing file data. CephFS is not specific to Proxmox. However, in Proxmox environments when you configure a Ceph storage pool, it uses the same file system that Proxmox uses for writing file data blocks and keeping replica data for resiliency from top to bottom.

CephFS can handle vast amounts of file metadata and data and be installed on commodity virtualization hardware. It is an excellent solution for many use cases, especially when integrated with a Ceph storage cluster, as we can do in Proxmox.

CephFS configuration in Proxmox: An Overview of the lab

After you have a working Ceph cluster on top of a Proxmox installation, including Ceph mgr, cluster monitors (Ceph mon), Ceph OSDs, daemons, cluster network, and a Ceph storage pool, how do you enable the Ceph file system on top of that? It is super easy to do in Proxmox, especially since everything is integrated. We will see this integration in the menus below.

As a note, in this example I am running Proxmox VE version 8.1.3 and Ceph Quincy, which are the latest updates to the platform from the official site with various security enhancements and features. For the lab, I am running a simple 4 node member cluster (started with 3 but was doing other testing and added a node) in nested virtual machines on an SSD disk with 3/2 Crush rule. You can configure different rules based on your needs and infrastructure.



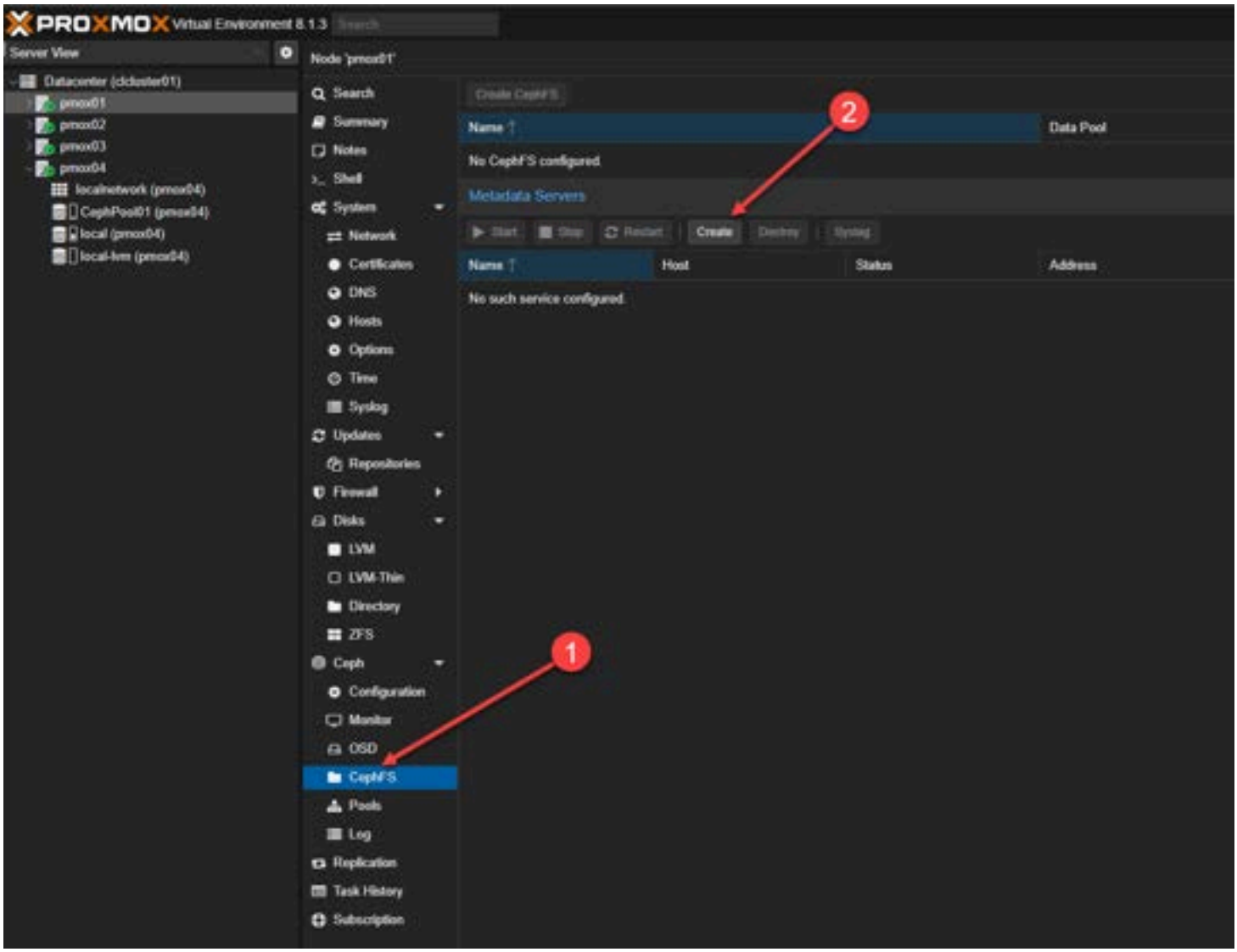
Cephfs home lab in proxmox

I set to replicated rule and a single NIC (multiple NICs and networks are recommended) for each machine running pveceph. In this small configuration, it leads to a significant amount of space used with replicas taking up 75% of the

capacity in order to created the replicated data and additional writes with changes.

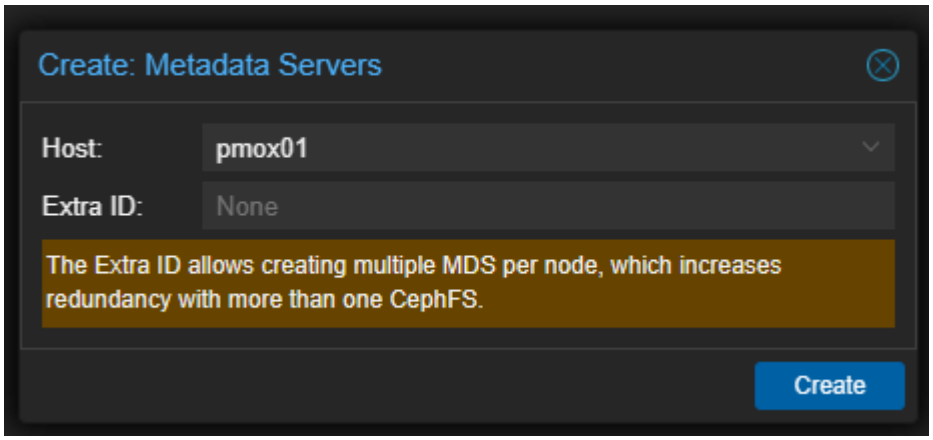
Installation steps

First, click the **CephFS** menu under Ceph for your Proxmox host. Next, you click the **Create** button in the Proxmox web app.



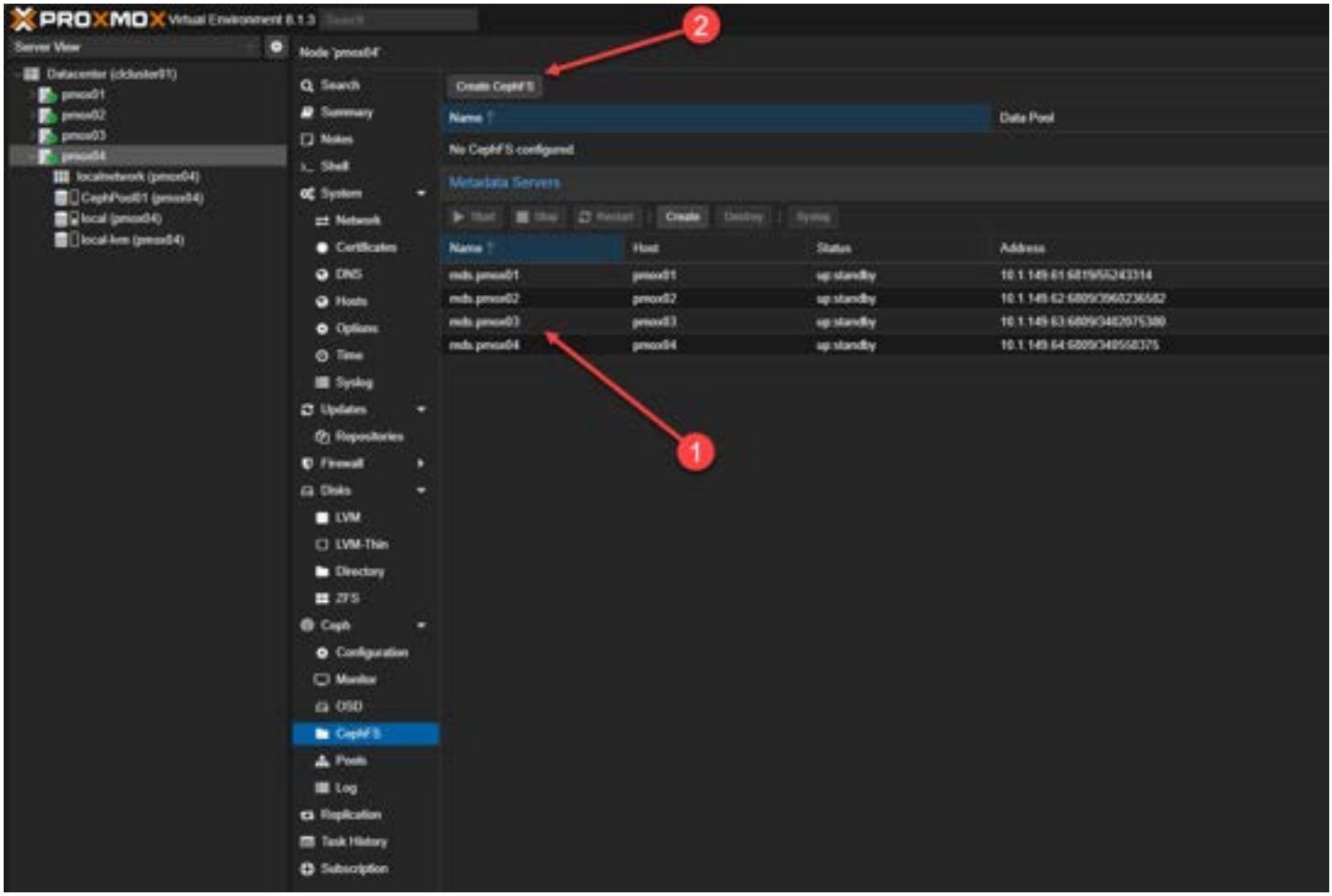
Beginning to create cephfs in proxmox

This will launch the dialog box to **Create: Metadata Servers**.



Create metadata servers dialog box

1) In my lab, I made each Proxmox host a Metadata server. 2) Click the **Create CephFS** button at the top.

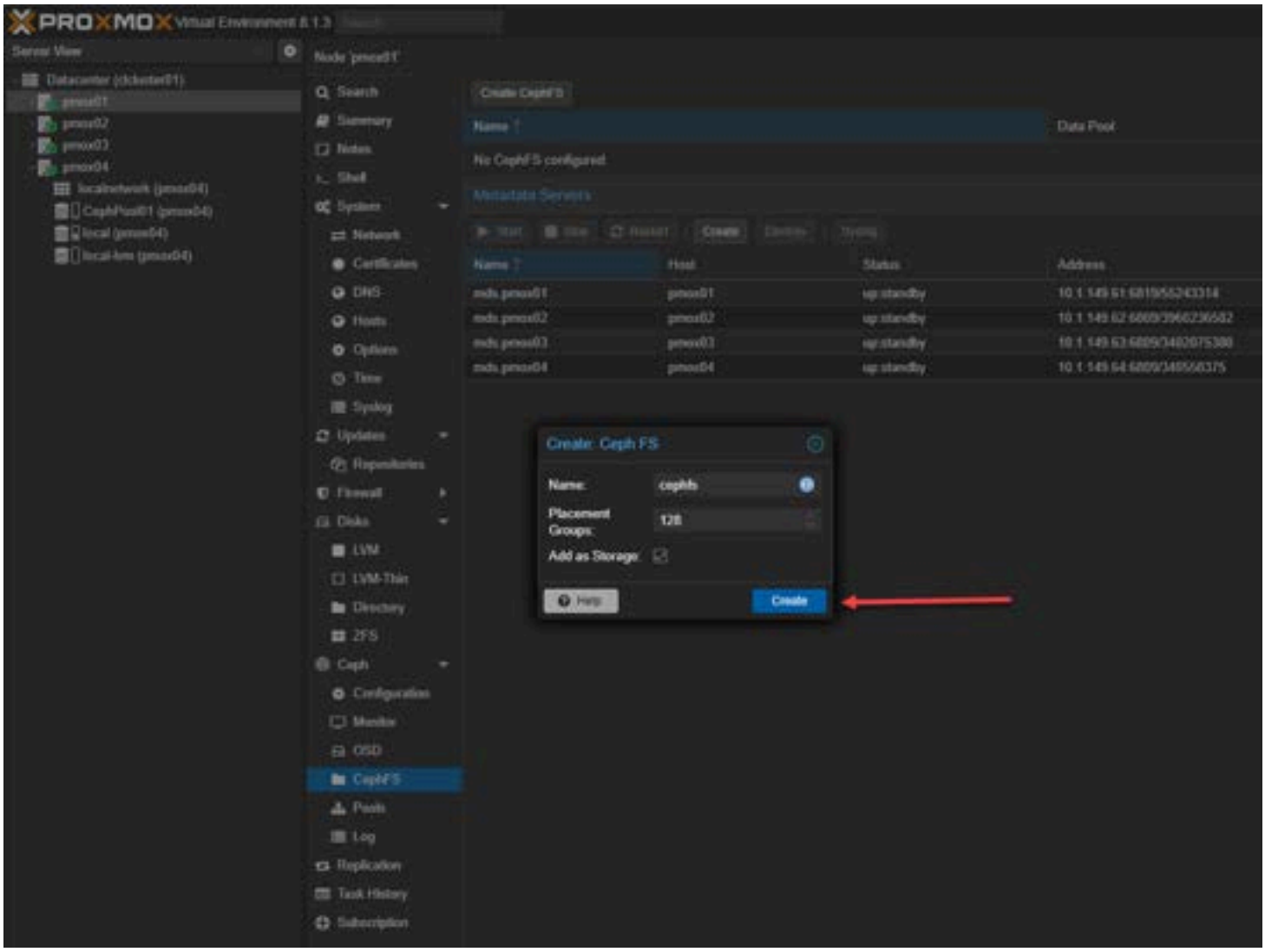


Click the create cephfs button

I left all the default options here:

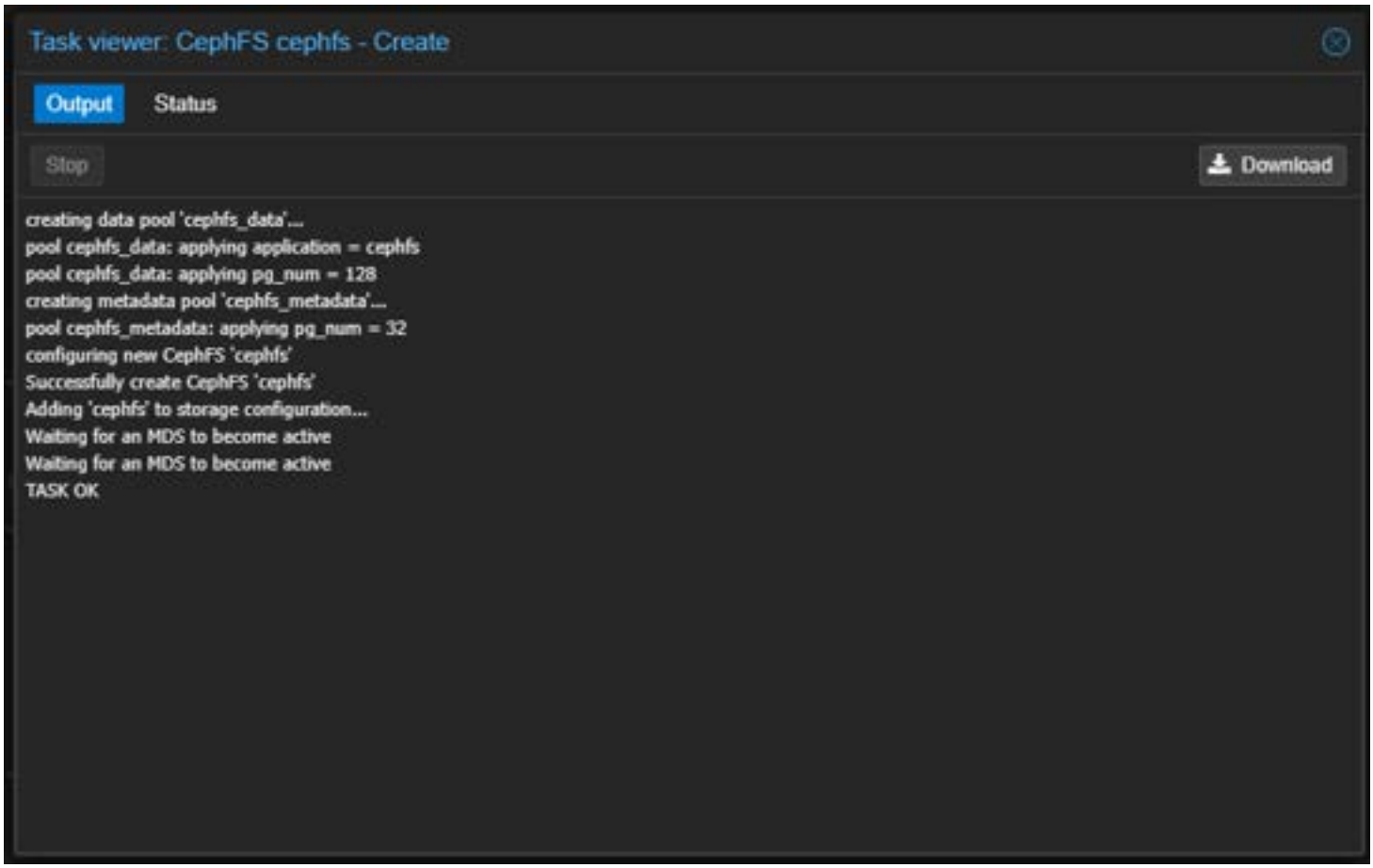
- Name
- Placement groups: default 128
- Add as Storage checked

Click **Create**.



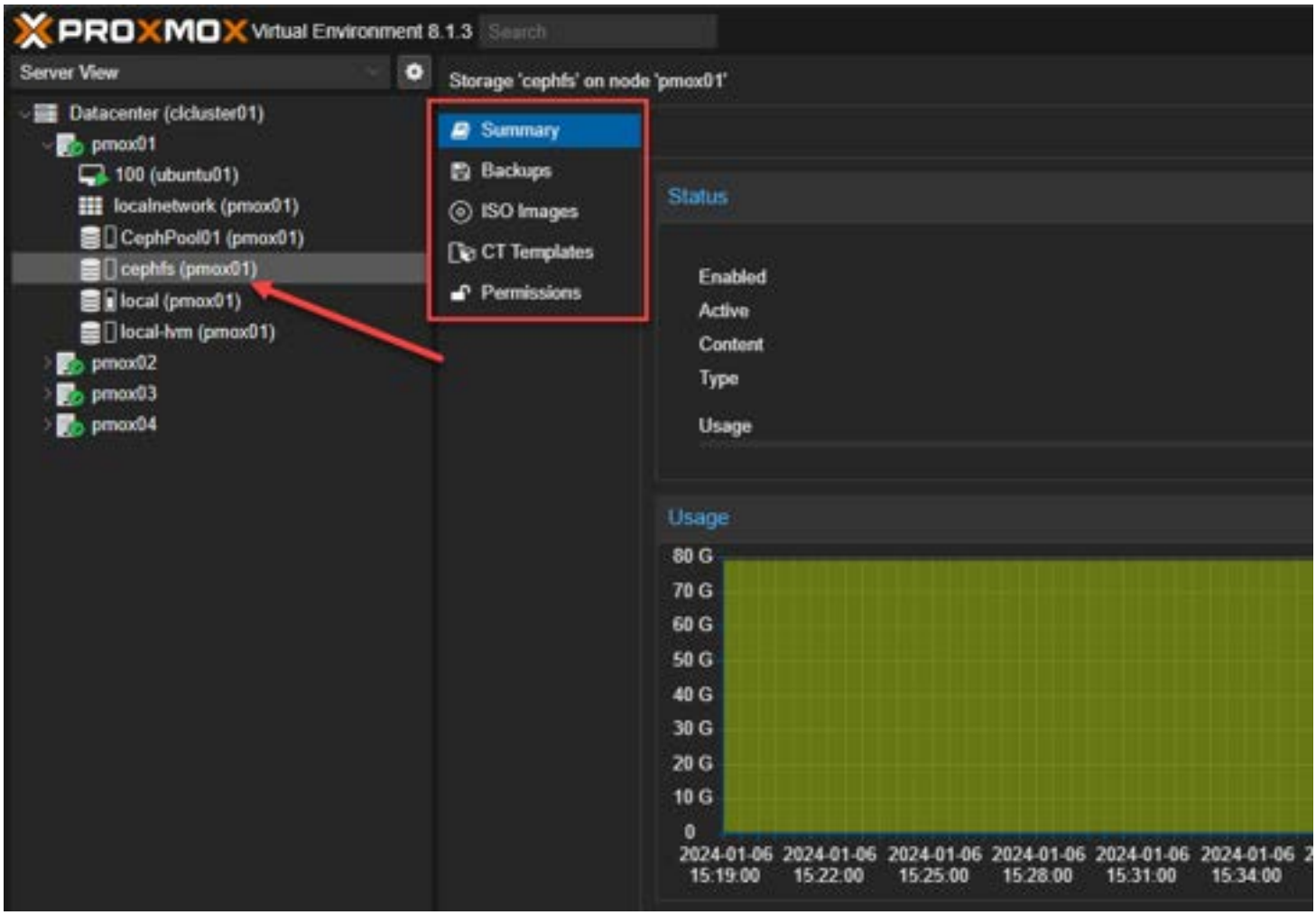
Create cephfs after creating metadata servers

In the Task viewer you will see the status of the task which should complete successfully.



Viewing the create cephfs task

If you choose to mount as storage, you will see the CephFS storage listed under your Proxmox host(s). Also, the great thing about the CephFS storage is you can use it to store things like ISOs, etc on top of your Ceph storage pools. Note in the navigation, we see the types of resources and content we can store, including ISO disks, etc.



Viewing the cephfs storage in proxmox

Installing Ceph client tools in Linux

To work with Ceph FS on Linux client nodes (Ceph clients, you install the Ceph client tools software packages from the CLI.sudo apt install ceph-common

```
linuxadmin@cldockertest3:~$ sudo apt install ceph-common
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libflashrom1 libftdi1-2
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  ibverbs-providers libbabeltrace1 libboost-context1.74.0 libboost-filesystem1.74.0 libboost-iostreams1.74.0
  liblua5.3-0 libndctl6 libnl-route-3-200 liboath0 libpmem1 libpmemobj1 librabbitmq4 librados2 libradosstripe
  python3-prettytable python3-rados python3-rbd python3-wcwidth
Suggested packages:
  ceph ceph-mds
The following NEW packages will be installed:
  ceph-common ibverbs-providers libbabeltrace1 libboost-context1.74.0 libboost-filesystem1.74.0 libboost-iost
  libibverbs1 liblua5.3-0 libndctl6 libnl-route-3-200 liboath0 libpmem1 libpmemobj1 librabbitmq4 librados2 li
  python3-cephfs python3-prettytable python3-rados python3-rbd python3-wcwidth
0 upgraded, 32 newly installed, 0 to remove and 11 not upgraded.
Need to get 35.0 MB of archives.
After this operation, 141 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y
Get:1 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libboost-iostreams1.74.0 amd64 1.74.0-14ubuntu3 [2
Get:2 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libboost-thread1.74.0 amd64 1.74.0-14ubuntu3 [262
Get:3 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libnl-route-3-200 amd64 3.5.0-0.1 [180 kB]
Get:4 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libibverbs1 amd64 39.0-1 [69.3 kB]
Get:5 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 librdmacm1 amd64 39.0-1 [71.2 kB]
Get:6 http://gb.archive.ubuntu.com/ubuntu jammy-updates/main amd64 librados2 amd64 17.2.6-0ubuntu0.22.04.2 [3
Get:7 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libdaxctl1 amd64 72.1-1 [19.8 kB]
Get:8 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libndctl6 amd64 72.1-1 [57.7 kB]
Get:9 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libpmem1 amd64 1.11.1-3build1 [81.4 kB]
Get:10 http://gb.archive.ubuntu.com/ubuntu jammy/main amd64 libpmemobj1 amd64 1.11.1-3build1 [124 kB]
Get:11 http://gb.archive.ubuntu.com/ubuntu jammy-updates/main amd64 librbd1 amd64 17.2.6-0ubuntu0.22.04.2 [3
```

Installing ceph common components on a linux client

Ceph fuse

Also, you can install the **ceph fuse** package. The ceph-fuse package is an alternate way of mounting CephFS. The difference is it mounts it in the userspace. The performance of ceph-fuse is not as good as the more traditional mounting of a CephFS file system.

However, it does allow you to connect to a Ceph distributed file system from a user's perspective, without the need to integrate it deeply into the system's core.

You can specify which Ceph file system to connect to either through a command line option (-m) or by using a configuration file (ceph.conf). This tool mounts the Ceph file system at a designated location on your system.sudo apt install ceph-fuse

```
~# sudo apt install ceph-fuse
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libflashrom1 libftdi1-2
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
  ceph-fuse
0 upgraded, 1 newly installed, 0 to remove and 11 not upgraded.
Need to get 856 kB of archives.
After this operation, 2,669 kB of additional disk space will be used.
Get:1 http://gb.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 ceph-fuse amd64 17.2.6-0ubuntu0.22.04.2 [856 kB]
Fetched 856 kB in 1s (571 kB/s)
Selecting previously unselected package ceph-fuse.
(Reading database ... 110666 files and directories currently installed.)
Preparing to unpack .../ceph-fuse_17.2.6-0ubuntu0.22.04.2_amd64.deb ...
Unpacking ceph-fuse (17.2.6-0ubuntu0.22.04.2) ...
Setting up ceph-fuse (17.2.6-0ubuntu0.22.04.2) ...
Created symlink /etc/systemd/system/remote-fs.target.wants/ceph-fuse.target → /lib/systemd/system/ceph-fuse.target.
Created symlink /etc/systemd/system/ceph.target.wants/ceph-fuse.target → /lib/systemd/system/ceph-fuse.target.
Processing triggers for man-db (2.10.2-1) ...
Scanning processes...
Scanning candidates...
Scanning linux images...

Restarting services...
Service restarts being deferred:
 systemctl restart ModemManager.service
 systemctl restart docker.service
 systemctl restart unattended-upgrades.service

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@cldockertest3:~#
```

Installing ceph fuse components

Things you will need for your CephFS configuration in Proxmox

There are a few steps you will need to connect to your Promxox CephFS installation:

1. The admin keyring
2. The name of the Ceph file system
3. The monitor addresses of your CephFS servers
4. A ceph.config file

1. The admin keyring

To see the admin credentials that you need to mount the CephFS file system, you need to get your **key** from the **ceph.client.admin.keyring** file. To get this, run the command:cat /etc/pve/priv/ceph.client.admin.keyring

You will see the value in the **key** section of the file. Note the user is **admin** and not **root.**

```
10.1.149.58 - PuTTY
root@pmox01:~# cat /etc/pve/priv/ceph.client.admin.keyring
[client.admin]
        key = AQAgPphlUFChMBAA2OsC3bdQ54rFA+1yqqjGKQ==
        caps mds = "allow *"
        caps mgr = "allow *"
        caps mon = "allow *"
        caps osd = "allow *"
root@pmox01:~#
```

Viewing the admin key in proxmox for cephfs

2. The name of the Ceph file system

The next piece of information you need is the name of the Ceph file system. To get that, you can run this command on your Proxmox host:ceph fs ls

You will see the name of the file system. The default name is **cephfs**.

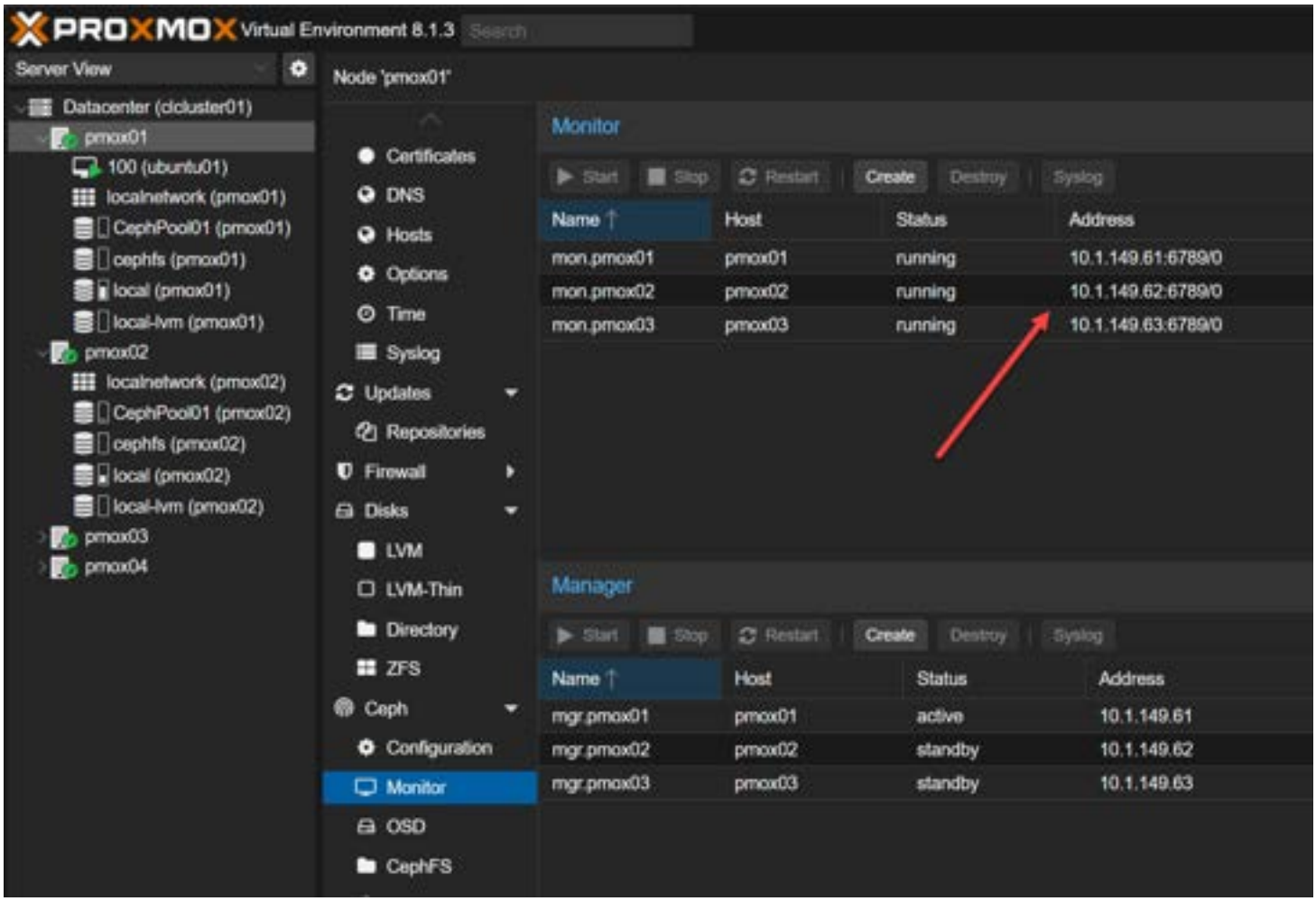
```
10.1.149.61 - PuTTY
root@pmox01:/# ceph fs ls
name: cephfs, metadata pool: cephfs_metadata, data pools: [cephfs_data ]
root@pmox01:/#
```

Running the ceph fs ls command

3. The monitor addresses of your Proxmox CephFS servers

You will need to have the Ceph monitor server addresses. There should be multiple servers configured as monitors for reliability and so you don't have a single point of failure.

You can file these hosts addresses under the **Ceph > Monitor** menu in the Proxmox GUI in the browser. Make sure your router or routers have the routes configured to allow your client devices to have connectivity to these IP addresses and port configurations.



Viewing the proxmox ceph monitor addresses

4. A ceph.config file

You will also need a **ceph.config** file. Like the admin keyring, we can also copy the file from the Promxox server. But we will trim some of the information out of the Proxmox server file. This file is located here on your Proxmox server:/etc/pve/ceph.config

Mine has the following contents for my Proxmox server environment.

```
[global]
     auth_client_required = cephx
     auth_cluster_required = cephx
     auth_service_required = cephx
     cluster_network = 10.1.149.61/24
     fsid = 75a2793d-00b7-4da5-81ce-48347089734d
     mon_allow_pool_delete = true
     mon_host = 10.1.149.61 10.1.149.63 10.1.149.62
     ms_bind_ipv4 = true
     ms_bind_ipv6 = false
     osd_pool_default_min_size = 2
     osd_pool_default_size = 3
     public_network = 10.1.149.61/24

[client]
     keyring = /etc/pve/priv/$cluster.$name.keyring

[mds]
     keyring = /var/lib/ceph/mds/ceph-$id/keyring

[mds.pmox01]
```

```
     host = pmox01
     mds_standby_for_name = pve

[mds.pmox02]
     host = pmox02
     mds_standby_for_name = pve

[mds.pmox03]
     host = pmox03
     mds_standby_for_name = pve

[mds.pmox04]
     host = pmox04
     mds_standby_for_name = pve

[mon.pmox01]
     public_addr = 10.1.149.61

[mon.pmox02]
     public_addr = 10.1.149.62

[mon.pmox03]
     public_addr = 10.1.149.63
```

Connect a Linux client to CephFS running on Proxmox

To connect a Linux client to our CephFS configuration in Proxmox, we need to create a couple of files. First, make the following directory:

```
mkdir /etc/ceph
```

In that directory create the files:

- admin.keyring
- ceph.conf

Running the tree command on the directory housing the cephfs configuration files

In the **admin.keyring** file, just put the **key** value in the file, nothing else. It will be a value as we had shown above that looks similar to this:

```
AQAgPphlUFChMBAA2OsC3bdQ54rFA+1yqqjGKQ==
```

Then, you will need the following in your **ceph.conf** file. As you can see below, I have updated the keyring location to point to our **admin.keyring** file.

```
[global]
auth_client_required = cephx
auth_cluster_required = cephx
auth_service_required = cephx
cluster_network = 10.1.149.0/24
fsid = 75a2793d-00b7-4da5-81ce-48347089734d
mon_allow_pool_delete = true
mon_host = 10.1.149.61 10.1.149.63 10.1.149.62
ms_bind_ipv4 = true
ms_bind_ipv6 = false
osd_pool_default_min_size = 2
osd_pool_default_size = 3
public_network = 10.1.149.0/24

[client]
keyring = /etc/ceph/admin.keyring
```

Run the mount command to mount the Ceph file system

We need to make a directory for the mount operation.

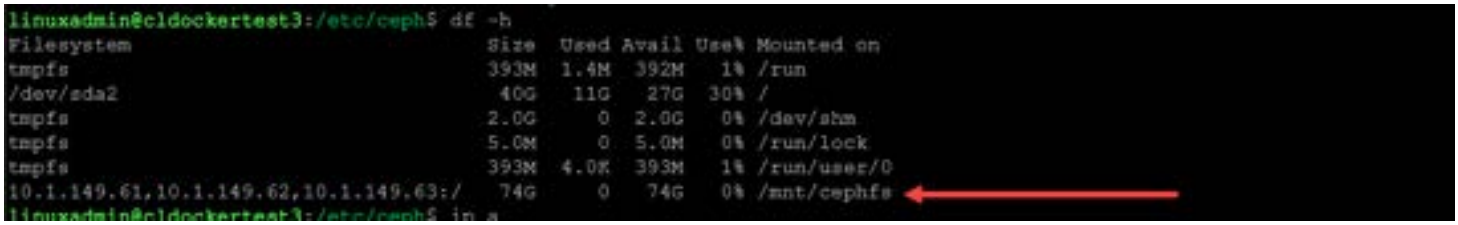
```
mkdir /mnt/cephfs
```

Now that we have a directory, we can run the following command to mount the CephFS file system for a connection to the IP address of each monitor node.

```
sudo mount -t ceph admin@75a2793d-00b7-4da5-81ce-48347089734d.cephfs=/ /mnt/cephfs -o
'secretfile=/etc/ceph/admin.keyring,mon_addr=10.1.149.61/10.1.149.62/10.1.149.63'
```

The command will complete without any return if it is successful. We can run the following to see our mounted Ceph file system:

```
df -h
```



Cephfs mounted in linux

Troubleshooting and support

Like any technology, there may be times when you need to troubleshoot something with CephFS. CephFS does not require a subscription license as it is free and open-source and can be pulled from the no-subscription repository.

Customers can of course, opt for enterprise support for your Proxmox cluster with the customer portal from the Proxmox team. If you still go the open-source route, the Proxmox support forum on the Internet is a great source of help for visitors across tens of thousands of threads thanks to activity members in the community. In addition, you can search forums for a wide variety of topics, instructions, question-answer type posts, etc.

There are a number of other home forums and websites, links, wiki sites and thread search titles where you can find people with experience to help with troubleshooting warning messages and errors, and share log data.

FAQs on CephFS configuration in Proxmox

How Does CephFS Support Object Storage in Proxmox?

CephFS is integrated with Proxmox and enhances object storage capabilities. It works alongside Ceph's RADOS Gateway (RGW) and allows storing and retrieving objects in separate RADOS pools. It enables both file and object storage.

Can CephFS Handle Erasure Coding for Data Protection?

CephFS supports erasure coding within its storage clusters. Erasure coding provides an efficient way to store data by breaking it up into chunks as opposed to traditional replication methods. It helps in large-scale deployments where data protection is of primary concern.

What Role Does the CRUSH Algorithm Play in CephFS?

The CRUSH algorithm specifies how data is stored across the cluster, enabling efficient distribution and availability. It allows scaling storage without compromising data access speed.

In Proxmox, How Does CephFS Ensure High Availability for Stored Data?

CephFS ensures high availability in Proxmox through its resilient cluster design. It replicates file data and metadata across different nodes. In node failures, the system automatically redistributes data to maintain access and integrity.

Are there any special considerations for CephFS in production environments?

When deploying CephFS in production environments, you need to make sure you have redundancy built-in with your cluster configuration, metadata servers, and Ceph monitors. Proper configuration helps maintain performance and stability in high-demand scenarios.

How Does CephFS Interact with External Monitoring Systems?

CephFS can be monitored with external monitoring systems. This can help provide insights into cluster health and performance. These systems can track metrics like storage utilization, I/O performance, and node status.

Does CephFS Support Snapshots and Clones in Proxmox?

CephFS fully supports snapshots and writable clones within Proxmox. This feature allows you to create point-in-time copies of files and directories, for data recovery and testing purposes.

What Is the Significance of Ceph MDS in a CephFS Setup?

It manages file system metadata, ensuring fast file and directory information access. MDS scales horizontally to handle increasing workloads, making it a key component in large-scale CephFS deployments.

What other products are made by Proxmox?

Proxmox also makes Proxmox Backup Server for protecting your Proxmox data as well as Proxmox Mail Gateway for mailflow services.

High Availability and Scalability in CephFS

CephFS filesystem inherits all the HA and scalability benefits of the Ceph storage pool. You can have multiple CephFS monitors, etc, in the case of a failure. These features allow the cluster to handle failed cluster nodes and leverage Ceph's distributed object store for data redundancy.

Ceph Block and CephFS

Understanding the Ceph storage cluster is crucial for optimal CephFS configuration. Ceph's distributed object store runs the file system CephFS services and provides a unified system for both file storage and block storage (vms), simplifying the configuration and providing HA and resiliency.

What are Metadata Servers in CephFS?

Metadata servers (MDS) in CephFS are responsible for storing and managing file metadata. This is important in the overall file system's performance. These servers allow efficient access and writing of file data blocks, for the Ceph file system's scalability and speed.

Wrapping up installing CephFS in Proxmox

CephFS configuration in Proxmox is an extremely powerful storage solution you can run in Proxmox for most dev and prod environments. However, not only does it allow you to have hyper-converged storage for vm instances and LXC container instances, it allows you to have file storage you can mount for clients that runs on top of the Ceph storage pool, with all the benefits that come with Ceph in terms of resiliency, scalability, and availability.

There are a lot of community support resources if you need help troubleshooting issues or figuring out the details and settings, even with the open-source no-subscription channels for Proxmox and Ceph Quincy.

Proxmox HA Cluster Configuration for Virtual Machines

January 12, 2024
Proxmox

Proxmox ha cluster configuration for virtual machines

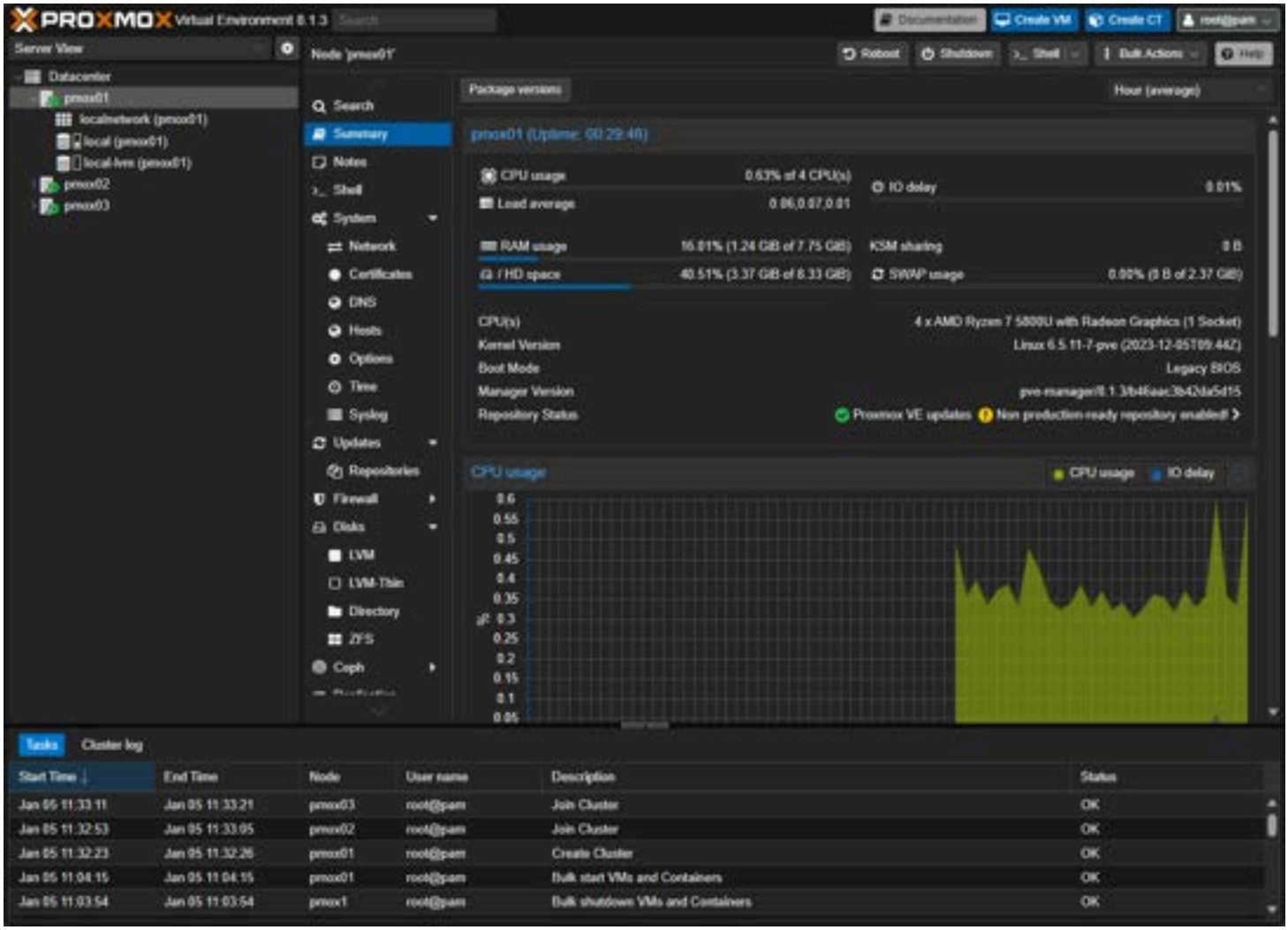
If you are learning the Proxmox hypervisor or want high-availability cluster resources for learning and self-hosting services with some resiliency, building a cluster is not too difficult. Also, you can easily create Proxmox HA virtual machine clustering once you create cluster nodes. Let's look at Proxmox HA virtual machine deployment and how to ensure your VM is protected against failure and increase uptime, much like VMware HA.

Table of contents

Proxmox cluster: the starting point

The starting point for a high availability solution with Proxmox is the Proxmox cluster. Most start with a single Proxmox server in the home lab. However, building a cluster requires a 2nd and third node. There are ways to increase the vote of one node if you have two Proxmox servers in a cluster if one goes down. However, for “production” Proxmox VE, having 3 nodes is the standard for configuring a minimum Proxmox clusters for scalability and avalability.

Below, I have three nodes in a Proxmox cluster running Proxmox 8.1.3 in the Proxmox UI.



Proxmox cluster running three nodes in an ha configuration

A Proxmox cluster includes multiple Proxmox servers or nodes that operate together as a logical unit to run your workloads. Understanding how to set up and manage the PVE cluster service effectively is important to ensure your VM data is protected and you have containers hardware redundancy.

Remember that this doesn’t replace all the other best practices with hardware configurations, such as redundant network hardware and power supplies in your Proxmox hosts and UPS battery backup as the basics.

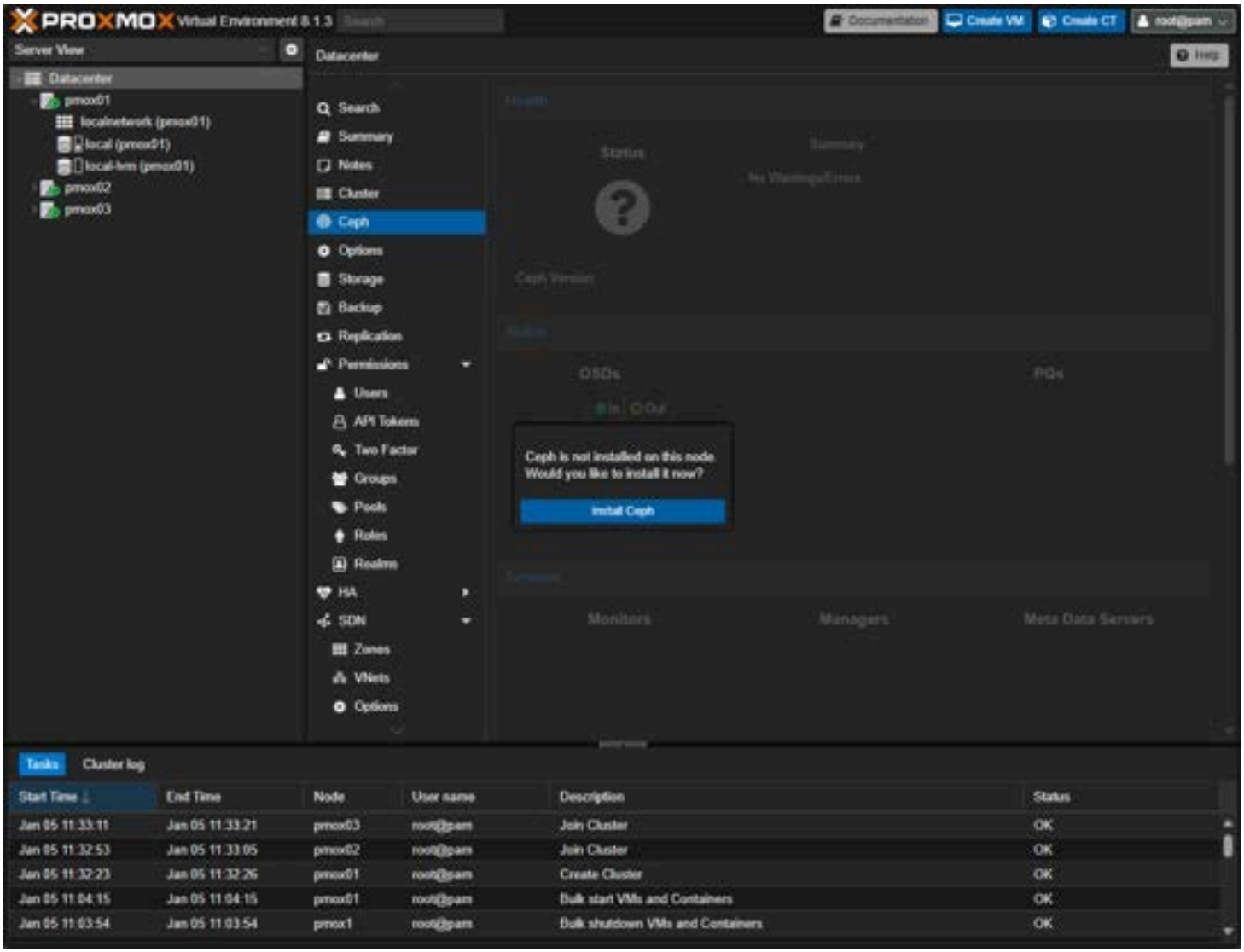
Shared storage

When you are thinking about a Proxmox cluster and virtual machine high availability, you need to consider integration with shared storage as part of your design. Shared storage is a requirement so that all Proxmox cluster hosts have access to

the data for your VMs. If a Proxmox host goes down, the other Promox hosts can pick up running the VM with the data they already have access to.

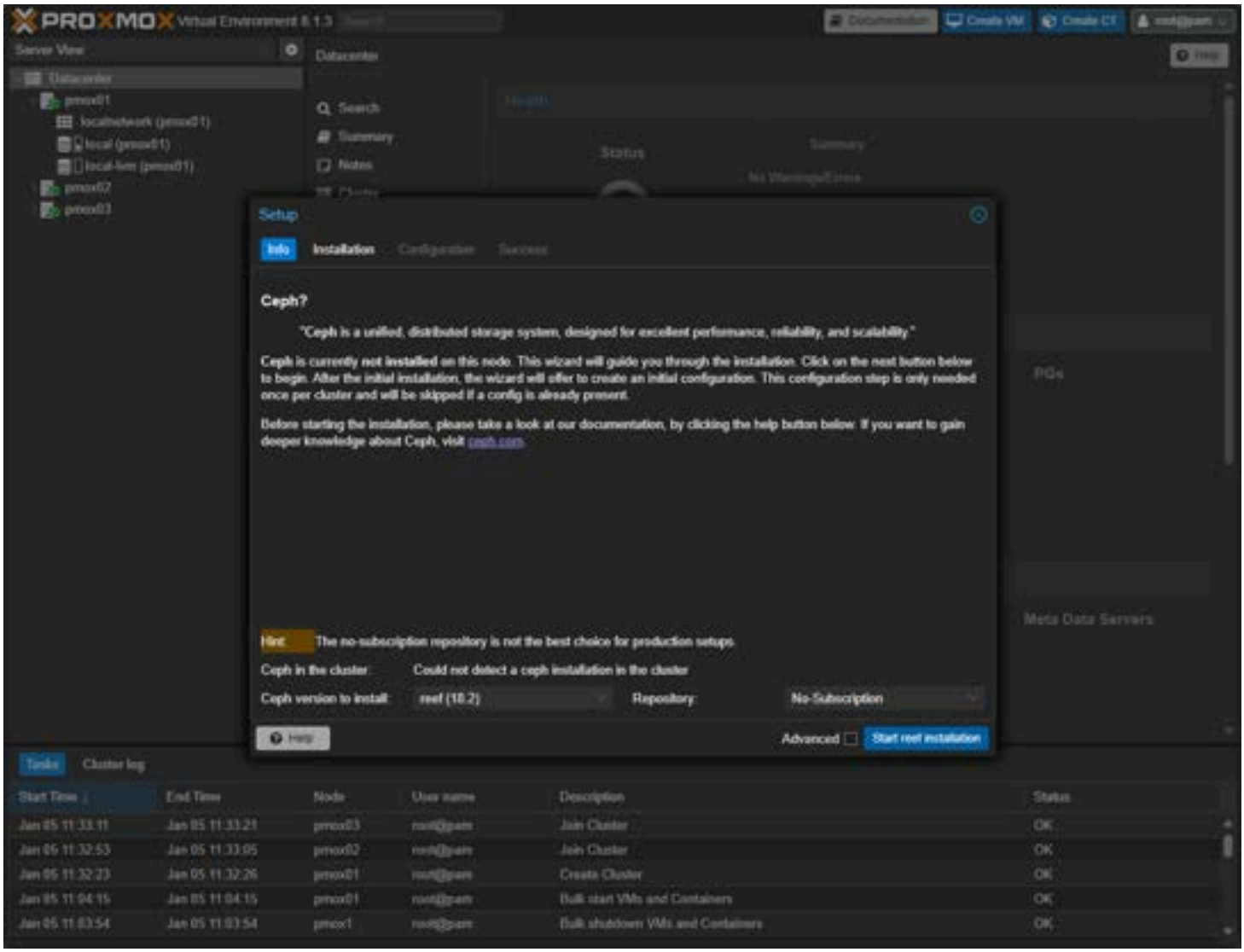
You can run a Proxmox cluster where each node has local storage, but this will not allow the VM to be highly available.

For my test cluster, I configured Proxmox Ceph storage. However, many other types of shared storage can work such as an iSCSI or other connection to a ZFS pool, etc. Below, we are navigating to Ceph and choosing to **Install Ceph**.



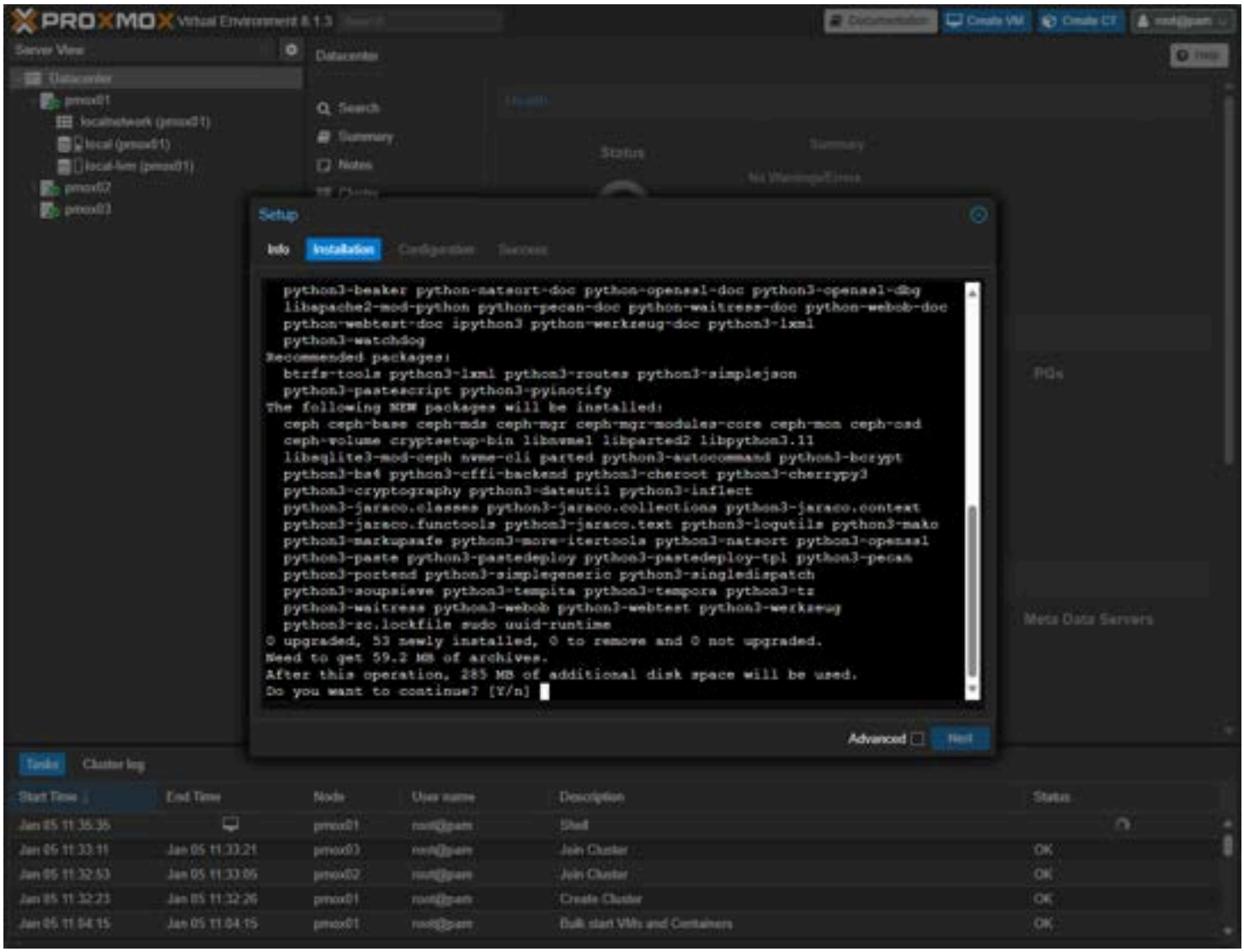
Getting started installing ceph

This launches the **Info** screen. Here I am choosing to install Ceph Reef and using the **No-Subscription** repo.



Starting the ceph setup

Type **Y** to begin the installation of Ceph.

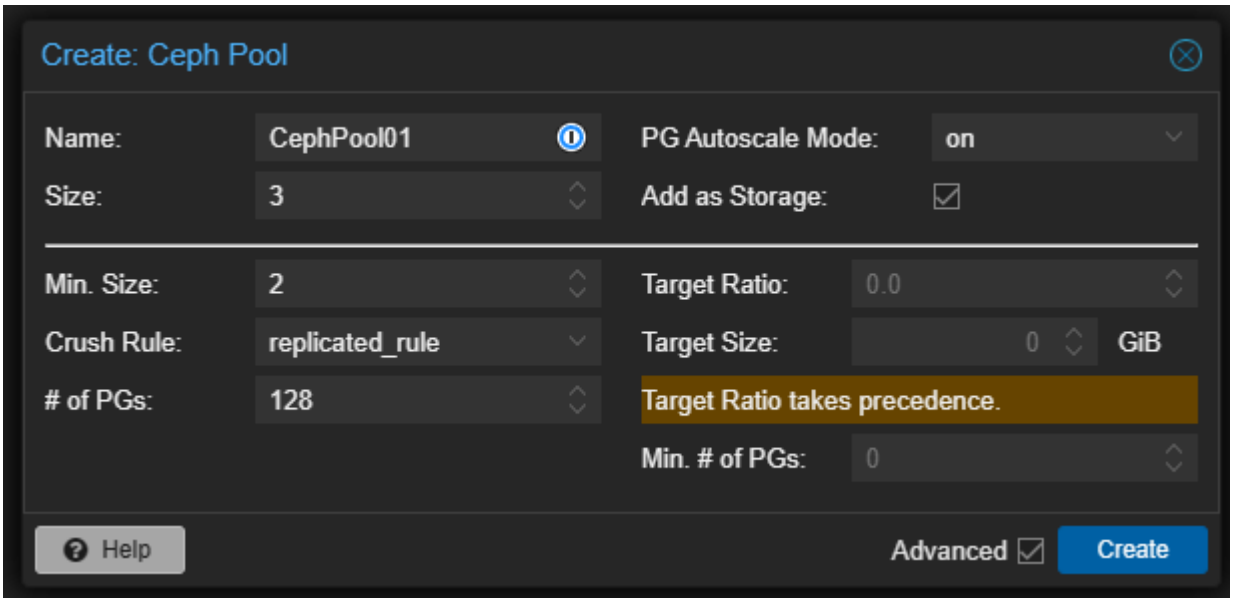


Confirm the ceph installation

Create the Ceph Pool, including configuring the:

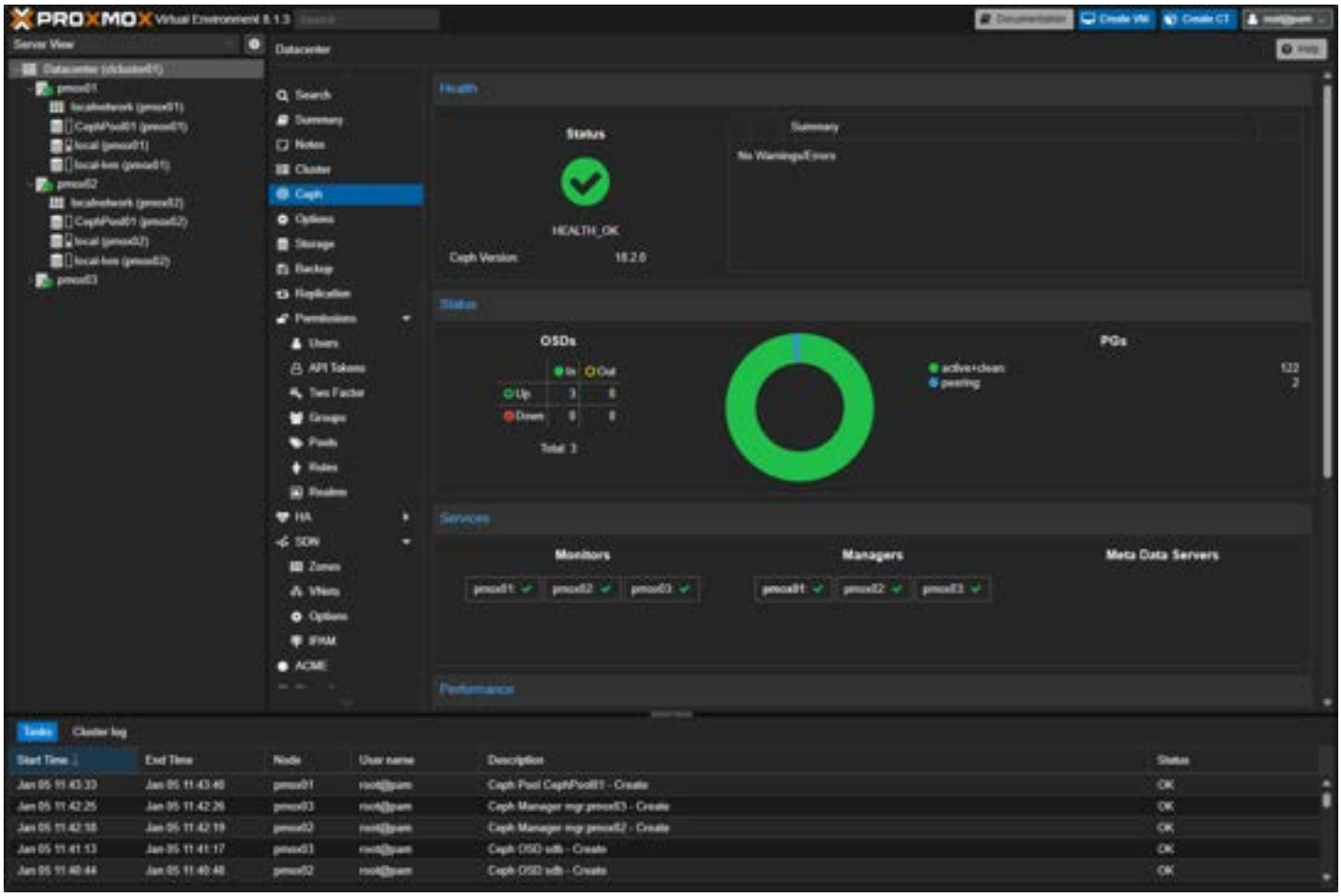
- Name
- Size
- Min Size
- Crush Rule
- # of PGs
- PG autoscale mode

Below is the default value for the configuration.



Creating the ceph pool

A healthy Ceph pool after installing Ceph on all three nodes, creating OSDs, Managers, Monitors, etc.



A healthy ceph storage configuration and pool

Setting Up Your Proxmox Cluster

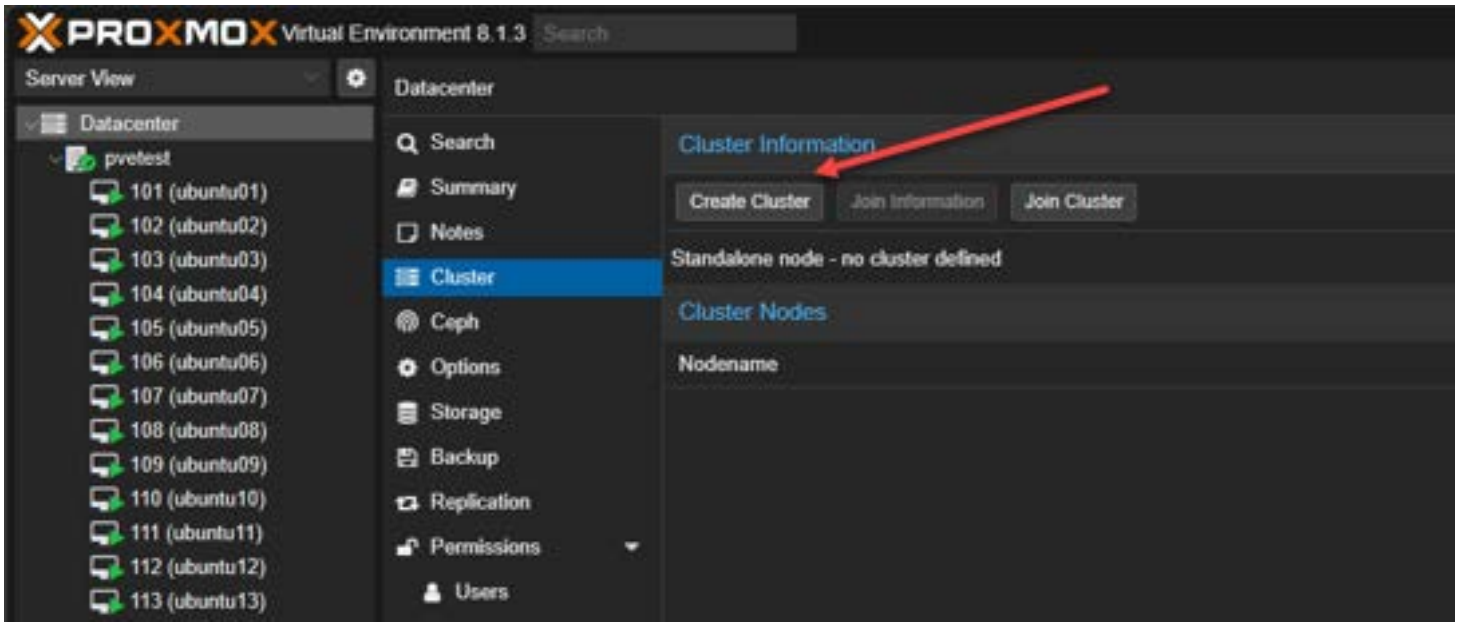
The journey to high availability begins with the creation of a Proxmox cluster. Here, we'll guide you through the process of joining multiple nodes to form a unified system. Each Proxmox node will contribute to the cluster's overall strength, offering

redundancy and reliability.

Key Steps in Creating a Proxmox Cluster

1. **Choosing Cluster Nodes**: Selecting the right Proxmox nodes is the first step. Ensure that each node is equipped with redundant network hardware and adequate storage capabilities.
2. **Configuring the Network**: A stable cluster network is vital. We'll explore how to set up a network that supports HA, focusing on IP address configuration and avoiding common pitfalls like split brain scenarios.
3. **Cluster Formation**: The process of forming a cluster involves initializing the first node and then adding additional nodes with the **join cluster** function. We'll walk you through the commands and steps necessary to create your cluster.

Let's look at screenshots of creating a Proxmox cluster and joining nodes to the cluster. Navigate to **Datacenter > Cluster > Create Cluster**.

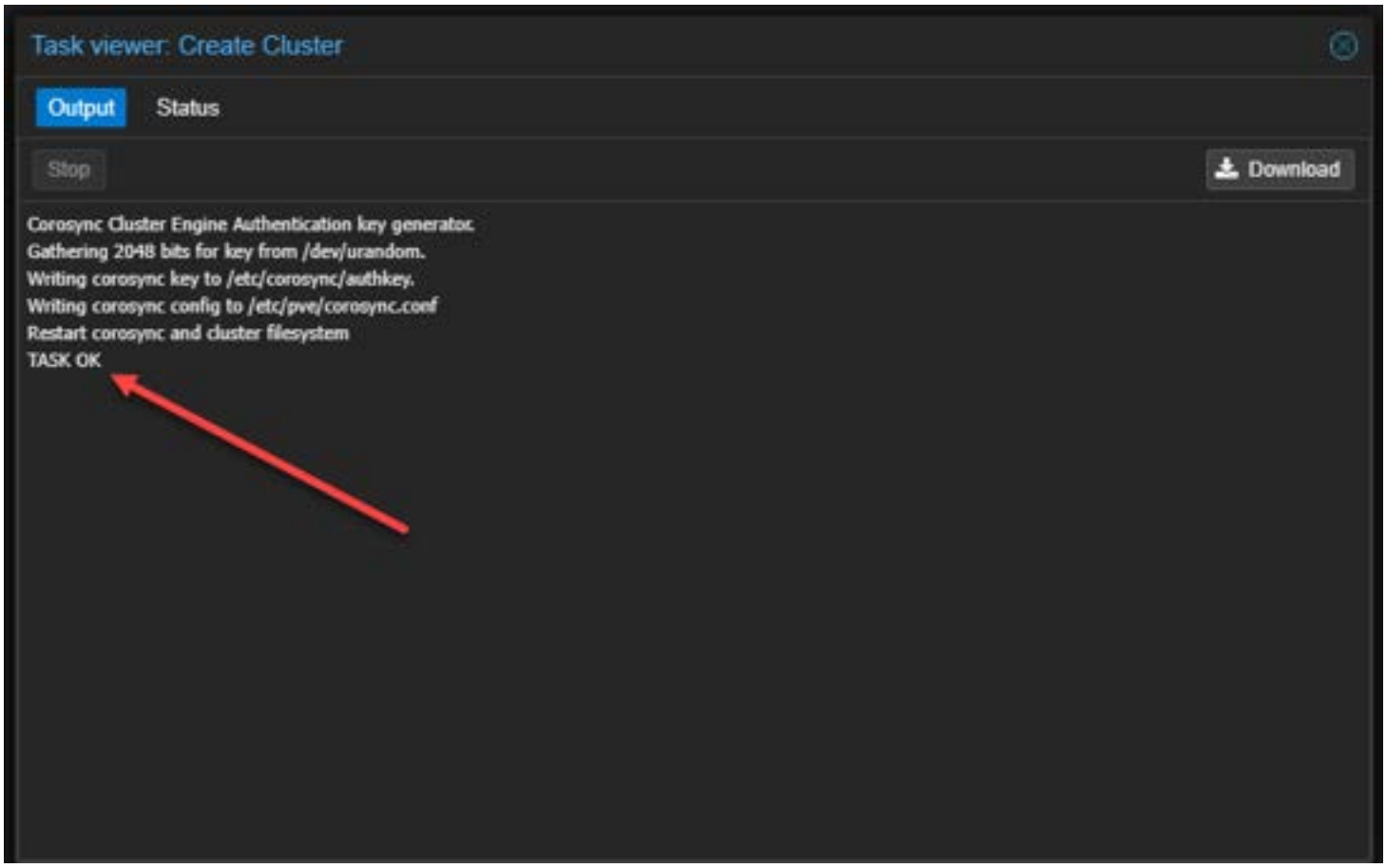


Beginning to create the cluster

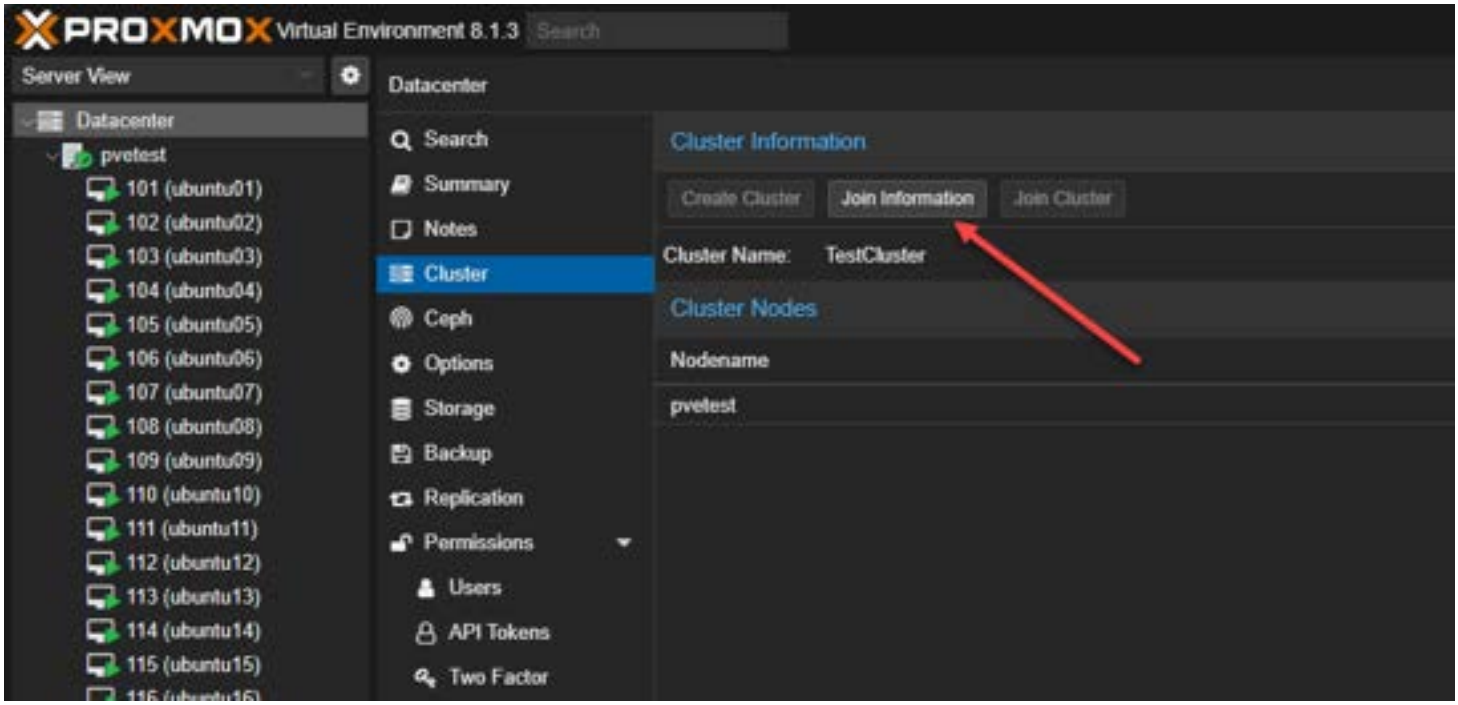
This will launch the **Create Cluster** dialog box. Name your cluster. It will default to your primary network link. You can **Add** links as a failover. Click **Create**.

Create the new cluster

The task will begin and should complete successfully.



The cluster creation completes successfully

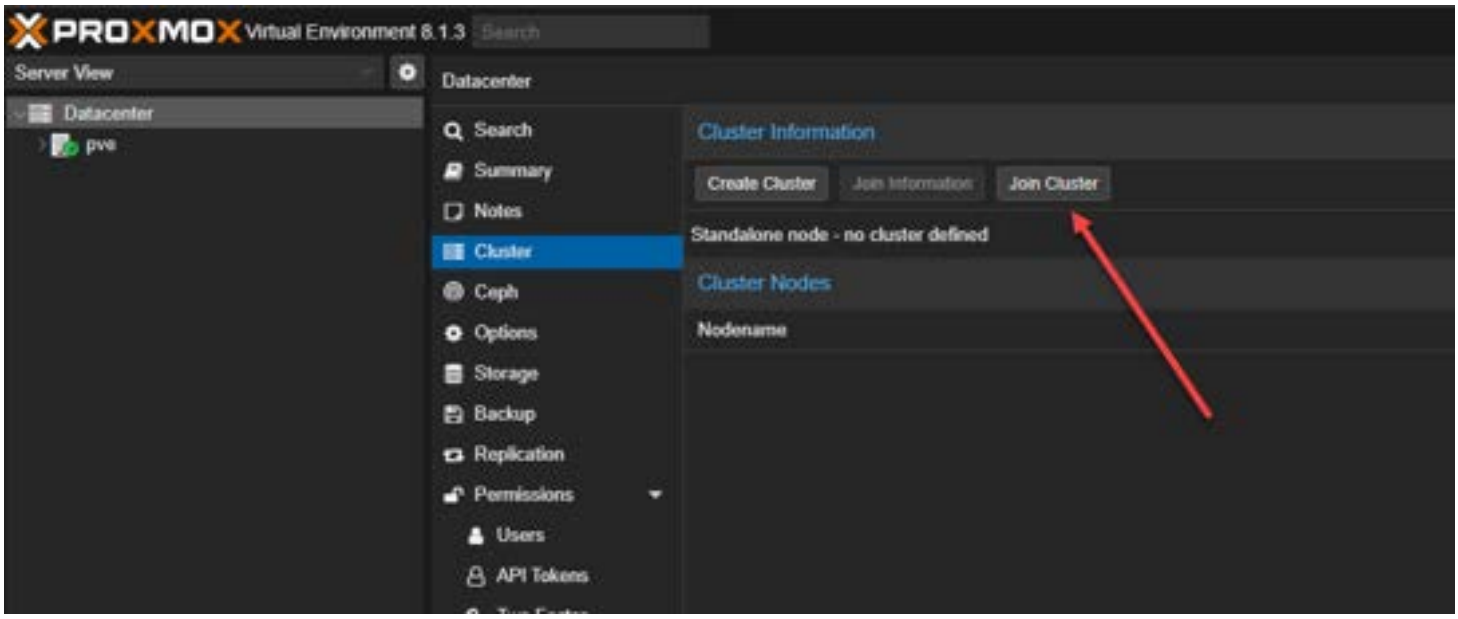


Get the cluster join information

You can then click the **Cluster join information** to display the information needed to join the cluster for the other nodes. You can click the copy information button to easily copy the join information to the clipboard.

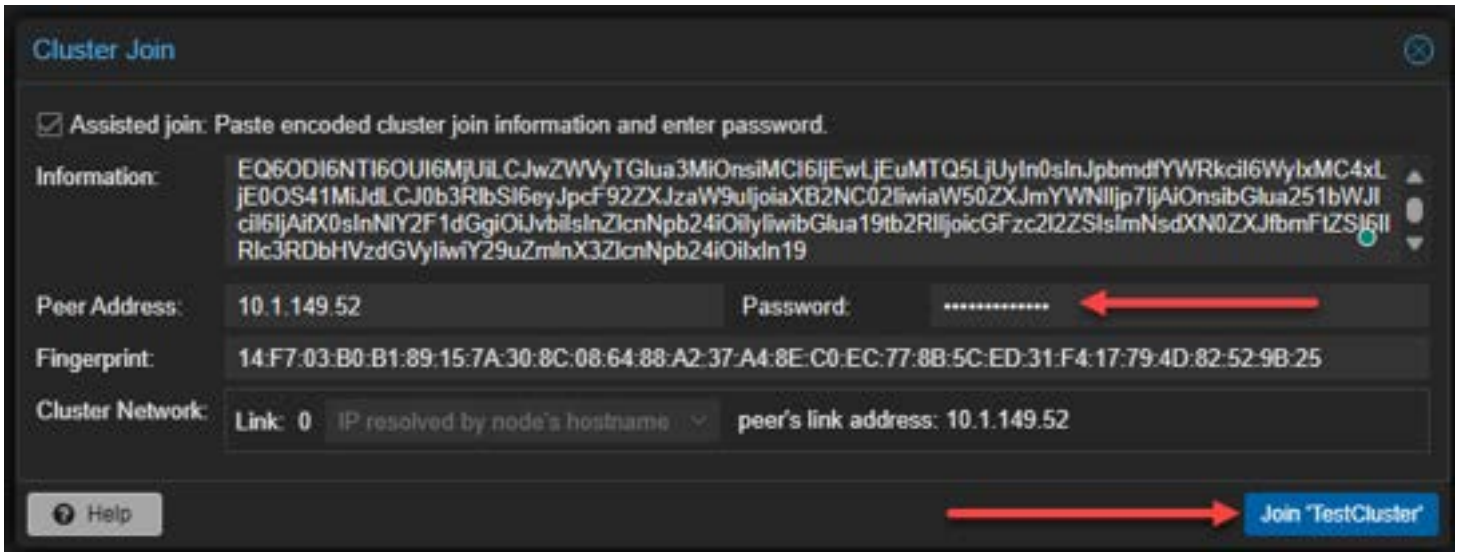
Viewing the join information

On the target node, we can click the **Join Cluster** button under the **Datacenter > Cluster** menu.



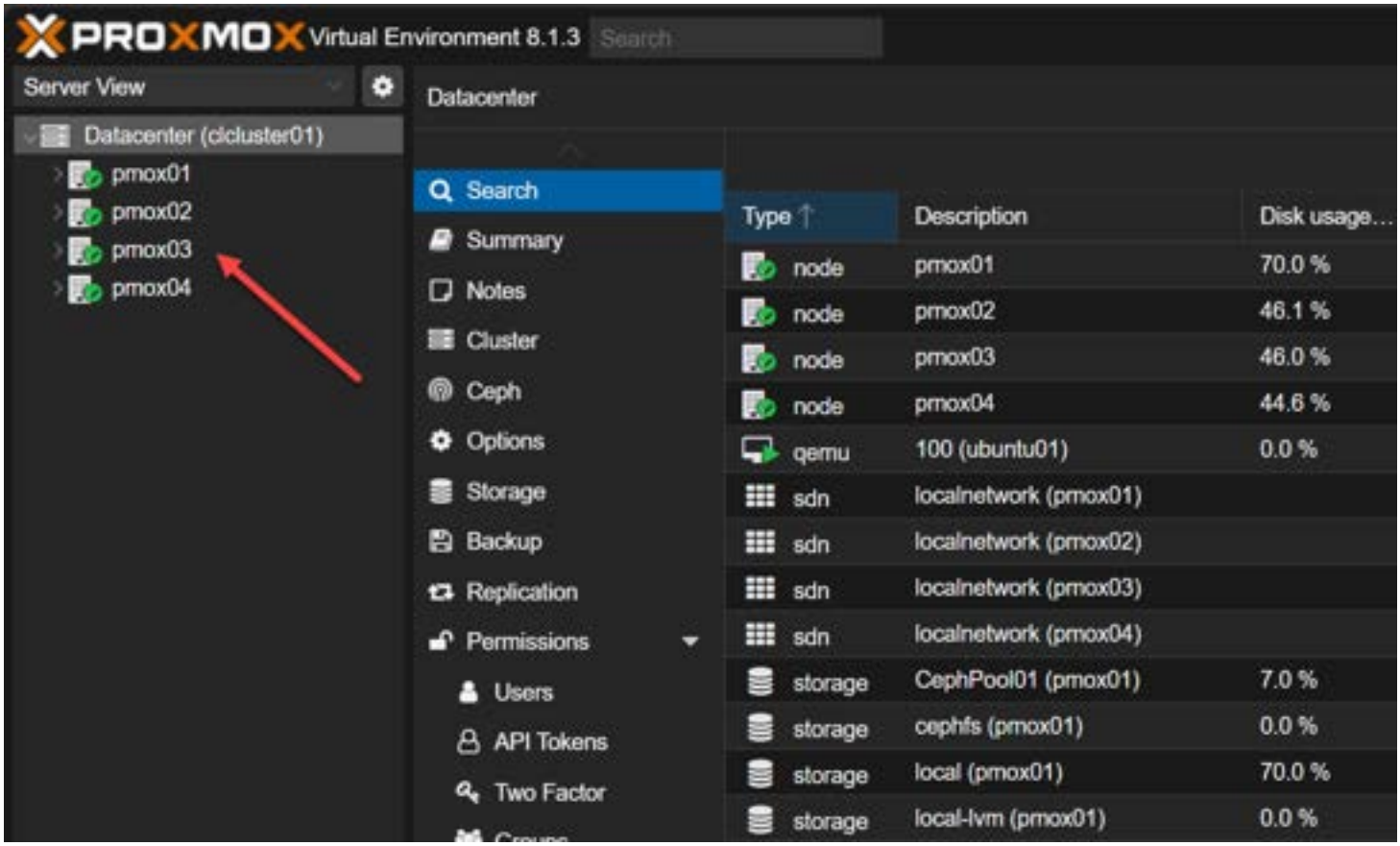
Join the cluster from another node

Now we can use the join information from our first node in the cluster to join additional nodes to the cluster. You will also need the root password of the cluster node to join the other Proxmox nodes.



Entering the join information

Below, I have created a cluster with 4 Proxmox hosts running Ceph shared storage.



A 4 node proxmox ve cluster

Configuring Virtual machine HA

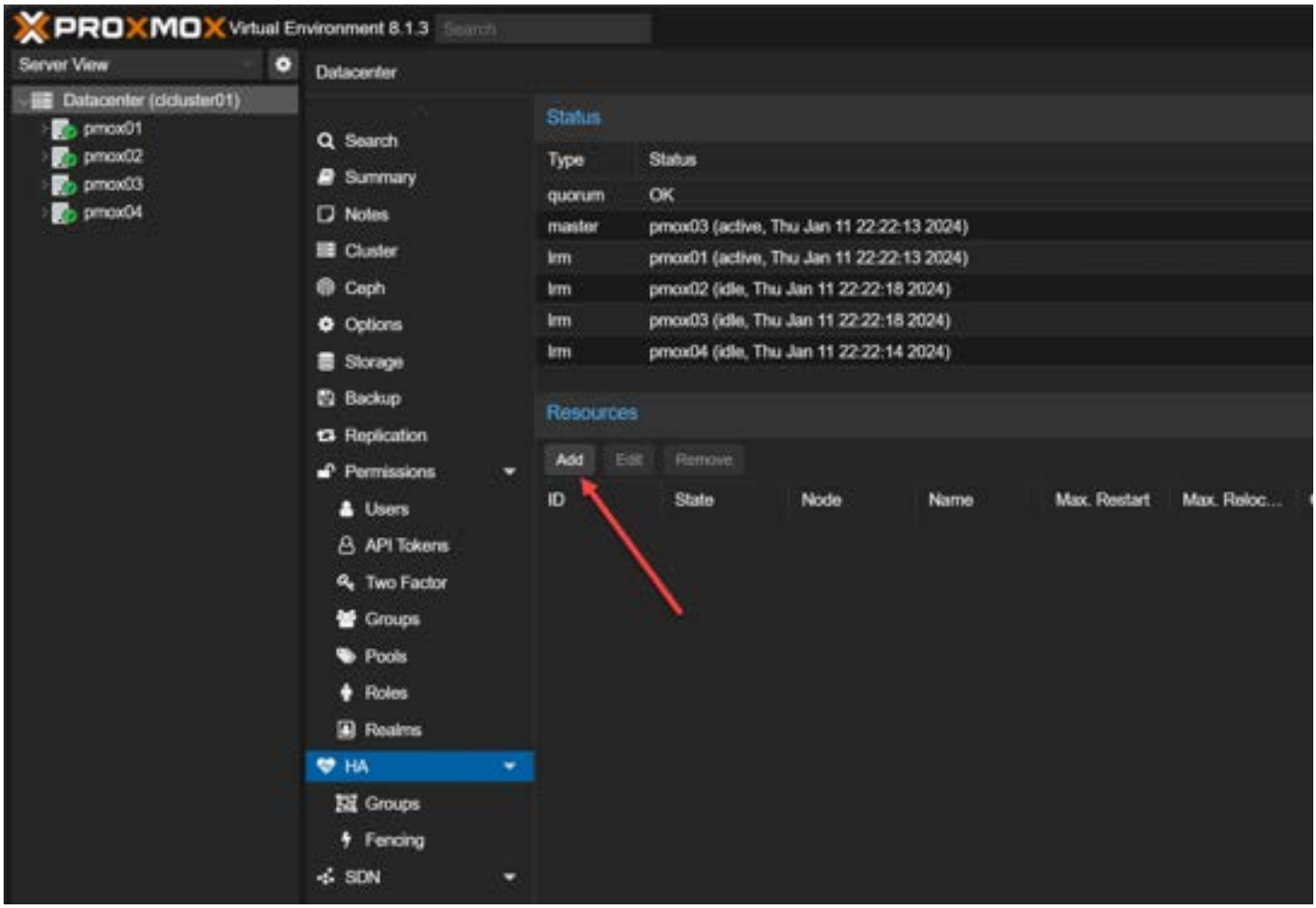
Once your Proxmox cluster is operational, the next step is configuring HA for your virtual machines. The Virtual Machine HA config provides automation for restarting a VM that is owned by a failed host, on a healthy host. This involves setting up shared storage, understanding HA manager, and defining HA groups administration.

High Availability Setup Requirements

When provisioning Proxmox high availability, there are a number of infrastructure requirements.

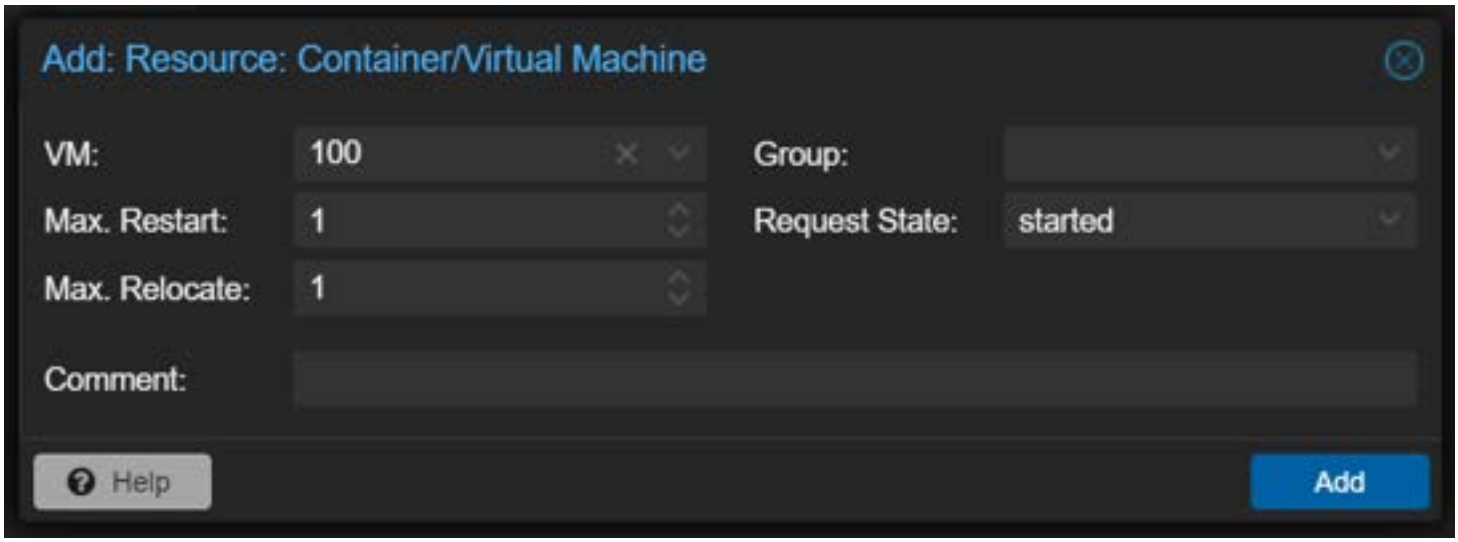
1. **Shared Storage Configuration**: For VMs to migrate seamlessly between nodes, shared storage is a necessity as we have mentioned above so data does not have to move during a failover.
2. **The HA Manager**: Proxmox's HA manager plays a critical role in monitoring and managing the state of VMs across the cluster. It works like an automated sysadmin. After you configure the resources it should oversee, such as VMs and containers, the ha-manager monitors their performance and manages the failover of services to another node if errors occur. Also, the ha-manager can process regular user commands, including starting, stopping, relocating, and migrating services.
3. **Defining HA Groups (optional)** : HA groups determine how VMs are distributed across the cluster.

Let's look at a basic example of configuring a single VM for high availability. Below, in the web interface we have navigated to the **Datacenter > HA > Resources > Add** button. Click the Add button.



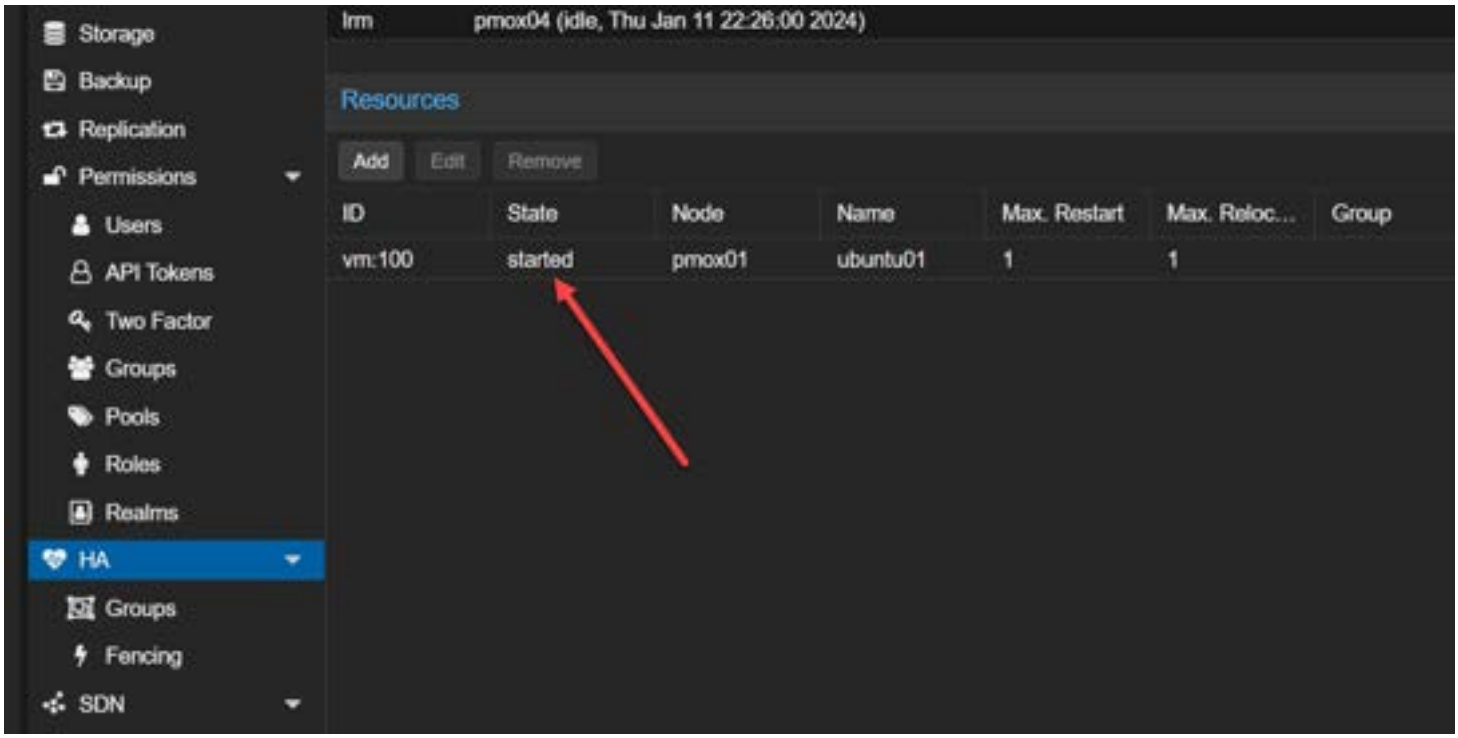
Add resources for ha

Select the VM ID to create the HA resource.



Add the resource id for the vm you want to be ha

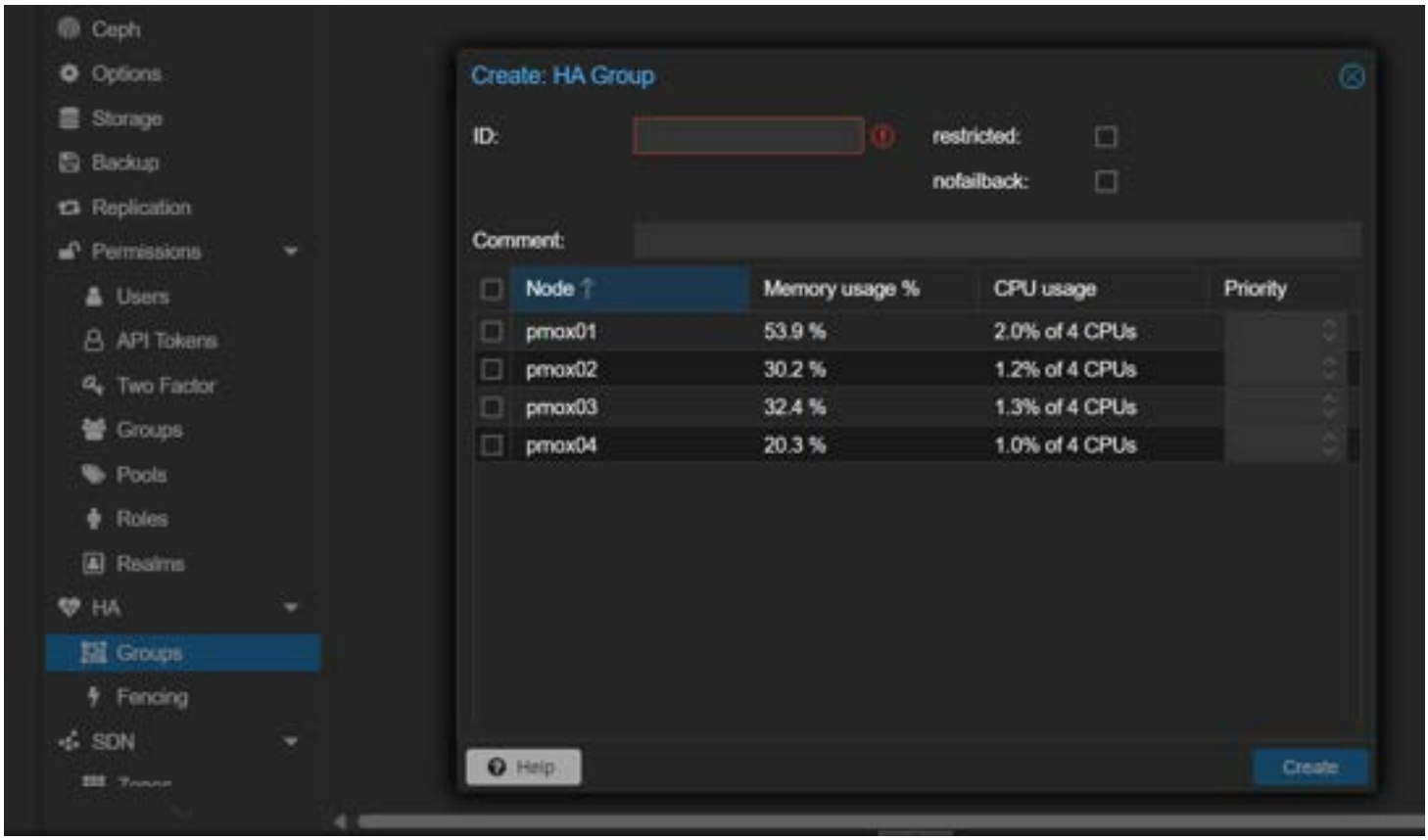
This will configure a service for the VM to make the VM highly available. The service will start and enter the **started** state. Now, we have the VM configured for HA.



The ha service for the vm is started

Configuring HA groups (optional)

The HA group configuration file /etc/pve/ha/groups.cfg defines groups of cluster nodes and how resources are spread across the nodes in the cluster. You can configure resources to only run on the members of a certain group. You can use this to give priority to certain VMs, on certain hosts. Below is the **Create HA Group** configuration dialog box.



Create an ha group for resources

Fencing device configuration

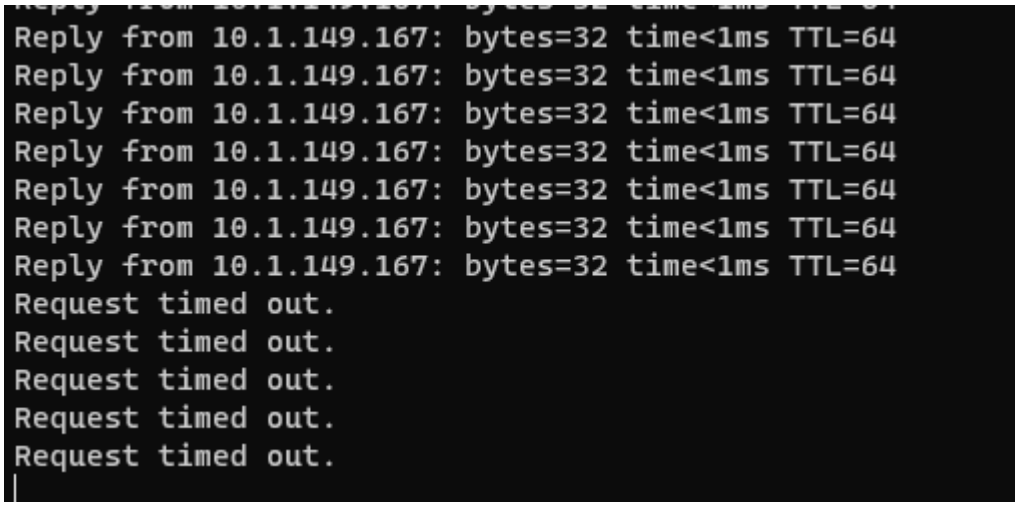
Fencing is important for managing node failures in Proxmox VE. When a host goes down and is completely offline, it prevents resource duplication during recovery. Without fencing, resources could run on multiple nodes simultaneously which can corrupt data.

Unfenced nodes can access shared resources, posing a risk. For instance, a VM on an unfenced node might still write to shared storage even if it's unreachable from the public network, causing race conditions and potential data loss if the VM is started elsewhere.

Proxmox VE employs various fencing methods, including traditional ones like power cutoffs and network isolation, as well as self-fencing using watchdog timers. These timers, integral in critical systems, reset regularly to prevent system malfunctions. If a malfunction occurs, the timer triggers a server reboot. Proxmox VE utilizes built-in hardware watchdogs on modern servers or falls back to the Linux Kernel softdog when necessary.

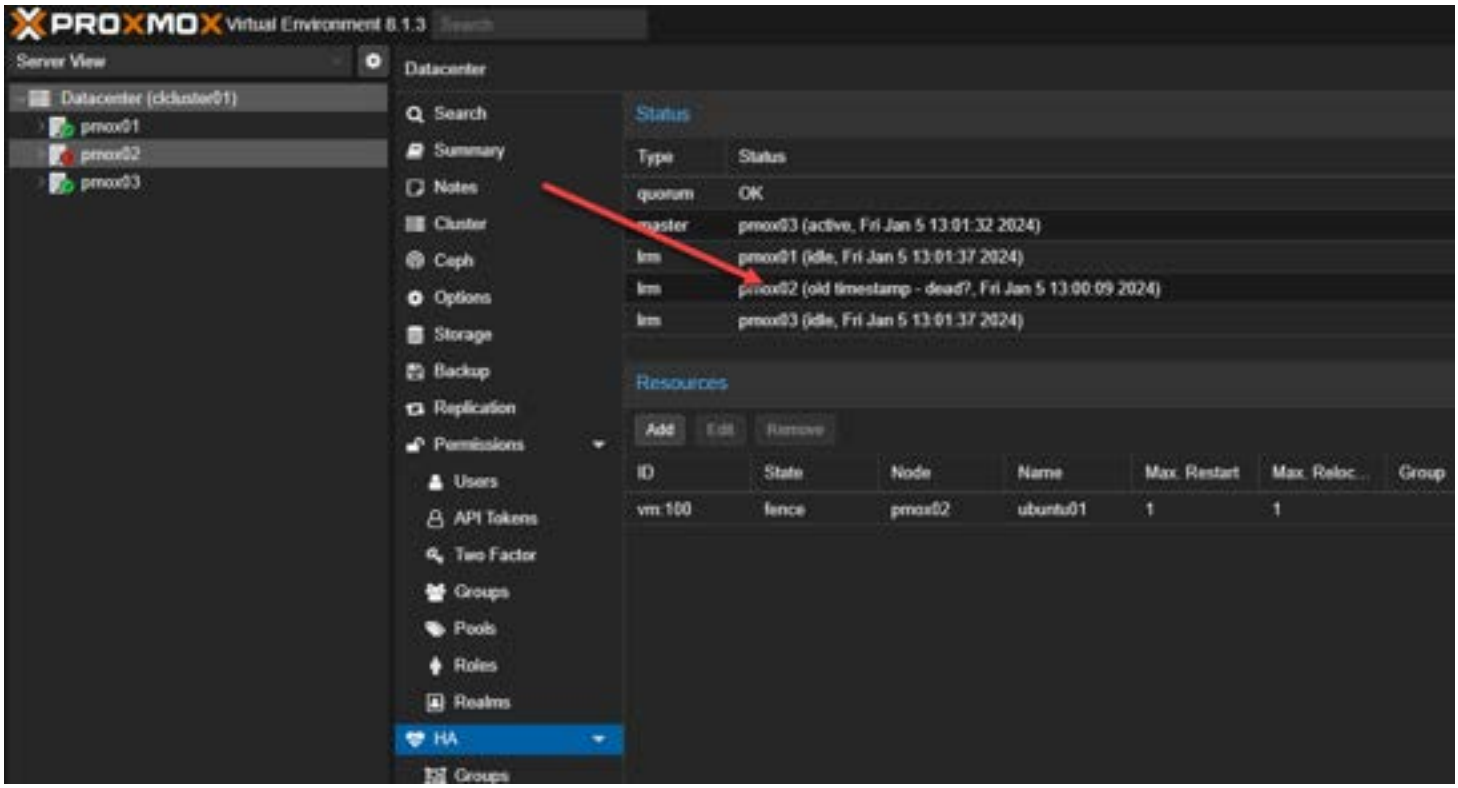
Fencing configuration in proxmox ve

Now, I simulated a failure of the Proxmox host by disconnecting the network connection. The pings to the VM start timing out.



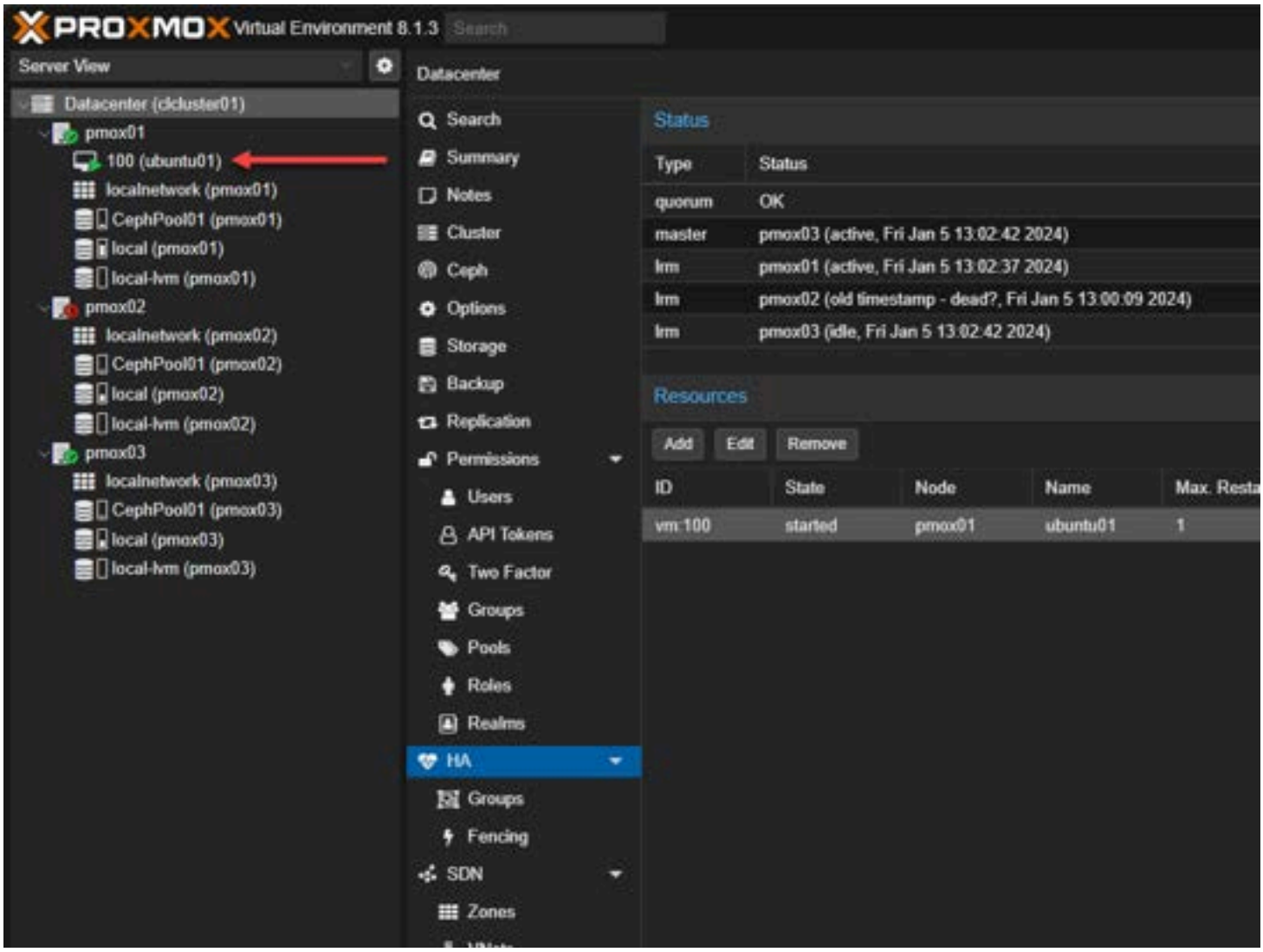
Virtual machine going down after a failed proxmox host

The host is now taking down the VM resource.



Proxmox ha viewing the server as dead

The HA process will restart the VM on a healthy host.



Virtual machine restarted on a healthy proxmox host

After just a couple of minutes, the VM restarts and starts pinging on a different host.

```
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.
Reply from 10.1.149.167: bytes=32 time=1ms TTL=63
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time=1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
Reply from 10.1.149.167: bytes=32 time<1ms TTL=64
```

The proxmox ha virtual machine configuration has brought the vm back up

Best Practices for Cluster Health

1. **Regular Updates and Backups**: Keeping your Proxmox servers and VMs up-to-date is critical. High availability is not a replacement for VM and container backup. Always protect your data with something like Proxmox Backup Server.
2. **Monitoring Tools and Techniques**: Proxmox has several tools for monitoring the health and performance of your cluster. Keep a check on your cluster health. Make sure monitor your nodes from the GUI and ensure things like shared storage are in a healthy state.
3. **Handling Node Failures**: Even with HA, node failures can happen. We'll cover the steps to recover from a failed node and how to ensure minimal impact on your virtual machines.
4. **Documentation**: Be sure to document the configuration of the cluster, including IPs, storage configuration, etc.

Rebooting Proxmox Servers running HA

If you want to reboot a Proxmox server for maintenance or other that is part of an HA cluster, you need to stop the following service on the node, either from the command line or GUI:

```
/etc/init.d/rgmanager stop
```

Frequently Asked Questions About Proxmox HA Configuration

Can I configure HA with just two Proxmox nodes?

Yes, it's possible to configure HA with two nodes, but it's not ideal due to the potential risk of split-brain scenarios. For optimal redundancy and reliability, a minimum of three nodes is recommended.

How does Proxmox handle VM migration in HA setups?

Proxmox automatically migrates VMs from a failed node to a functioning one within the cluster. This process is managed by the HA manager, which monitors node and VM states to initiate automatic failover.

What are the key considerations for Proxmox HA network configuration?

Key considerations include having a redundant network setup, ensuring reliable IP address allocation, and configuring a separate network for cluster communication to prevent data traffic interference.

Can I use local storage for Proxmox HA?

Local storage can be used, but it doesn't support live migration of VMs in case of node failure. Shared storage solutions like Ceph or NFS from a NAS as an option are preferred for true HA capabilities and settings.

What happens if the HA manager fails?

Proxmox's HA manager is designed with redundancy. If the primary manager fails and a change, another node in the cluster takes over its duties and goes into action, ensuring continuous monitoring and management of the HA setup.

How do I update VMs in a Proxmox HA cluster without downtime?

Use live migration to move VMs to another node in order to update the original node software. This ensures that your VMs remain operational during updates, minimizing downtime.

What is the role of a quorum in a Proxmox cluster?

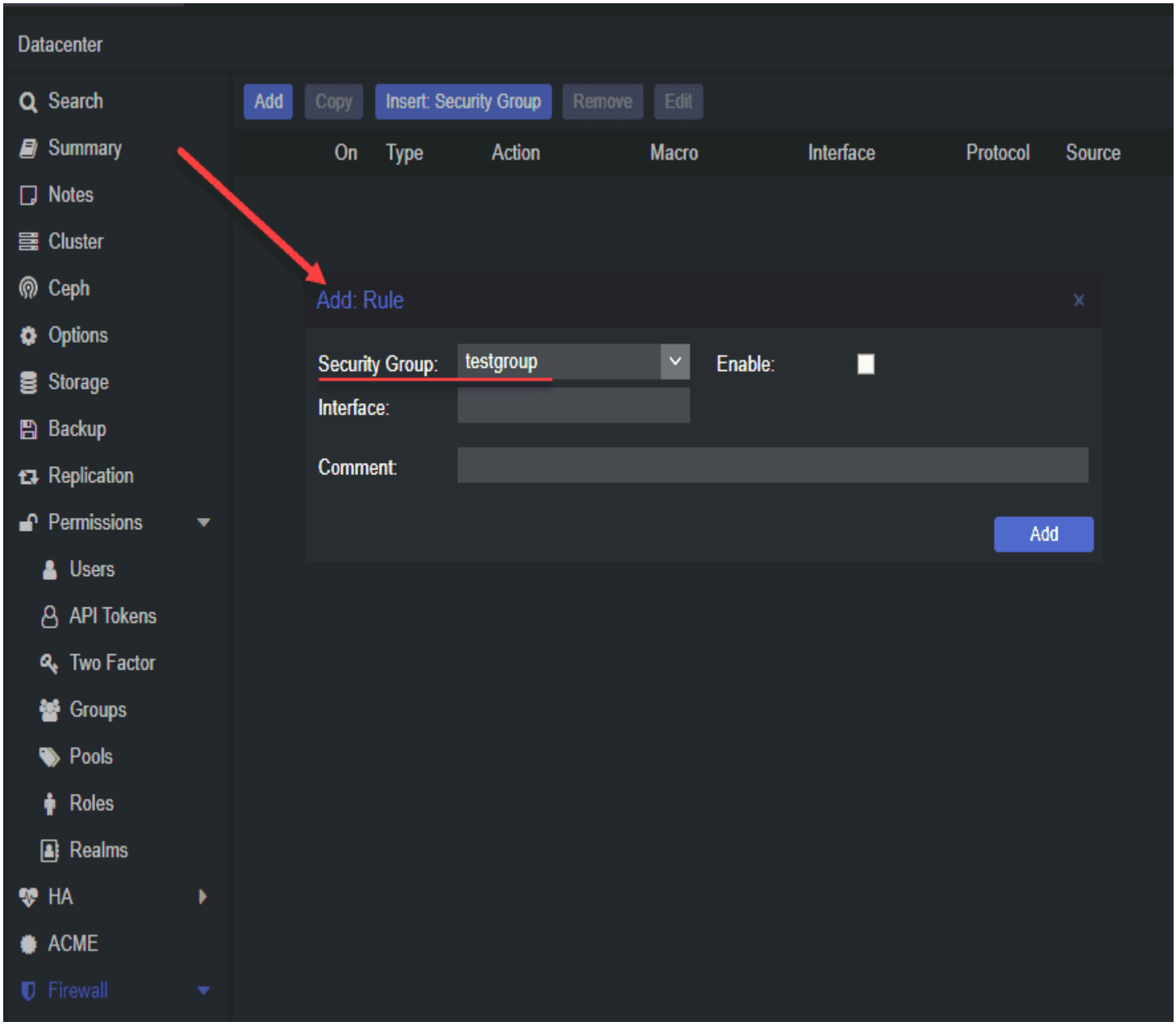
A quorum is used to ensure that decisions (like VM failover) are made reliably in the cluster. It prevents split-brain scenarios by requiring a majority of nodes to agree on the cluster state.

Wrapping Up

Proxmox virtualization has some great features, including high-availability configuration for virtual machines. In this article, we have considered the configuration of a high-availability Proxmox VE cluster and then configuring high availability for VMs. In the comments, let me know if you are running a Proxmox VE cluster, what type of storage you are using, and any other details you would like to share.

Proxmox firewall setup and configuration

March 2, 2023
Proxmox



8dc2e1d7 7e3e 44d8 9889 8f219e08a231

Proxmox VE is a great solution for home lab environments and production workloads, including virtual machines and containers. A great feature of Proxmox VE is the firewall, which enables administrators to manage network traffic to and from virtual machines and containers. This article will explore the Proxmox firewall and its configuration options.

Proxmox Firewall Rules Configuration

Proxmox firewall is based on the Linux iptables firewall. It is designed to filter network traffic at the hypervisor layer. With the firewall, you can filter traffic based on source and destination IP addresses, protocols, and ports.

Many management options exist, including the Proxmox web interface (web GUI) or command-line interface (CLI). These can be used to configure firewall rules and implement cluster-wide firewall configuration in your Proxmox cluster.

Zones configuration

You can divide the firewall into zones. This combines network interfaces and IP addresses. By default, notice the four zones available in Proxmox VE.

1. Input – handles incoming traffic from external networks
2. Output – handles outgoing traffic to external networks
3. Forward – handles traffic between virtual machines and containers
4. Host – handles traffic to and from the Proxmox host

Cluster Wide Firewall Rules

With clustered Promox configurations, you can have firewall rules to apply to all nodes in the cluster. This is done by configuring the underlying iptables rules automatically and using the same firewall configuration files on all nodes. You can also configure a central firewall solution for the entire cluster by creating a firewall cluster.

Proxmox VE Firewall Zones

To manage the firewall, you need to enable the firewall service. Once enabled, you can configure the firewall zones using the web interface or the command line.

You can also assign IP addresses to zones and create firewall rules that allow or block traffic based on the zone.

Enabling the Firewalls

The Proxmox firewall is disabled by default. To enable the firewall service, you can use the following command on the CLI:

pve-firewall enable

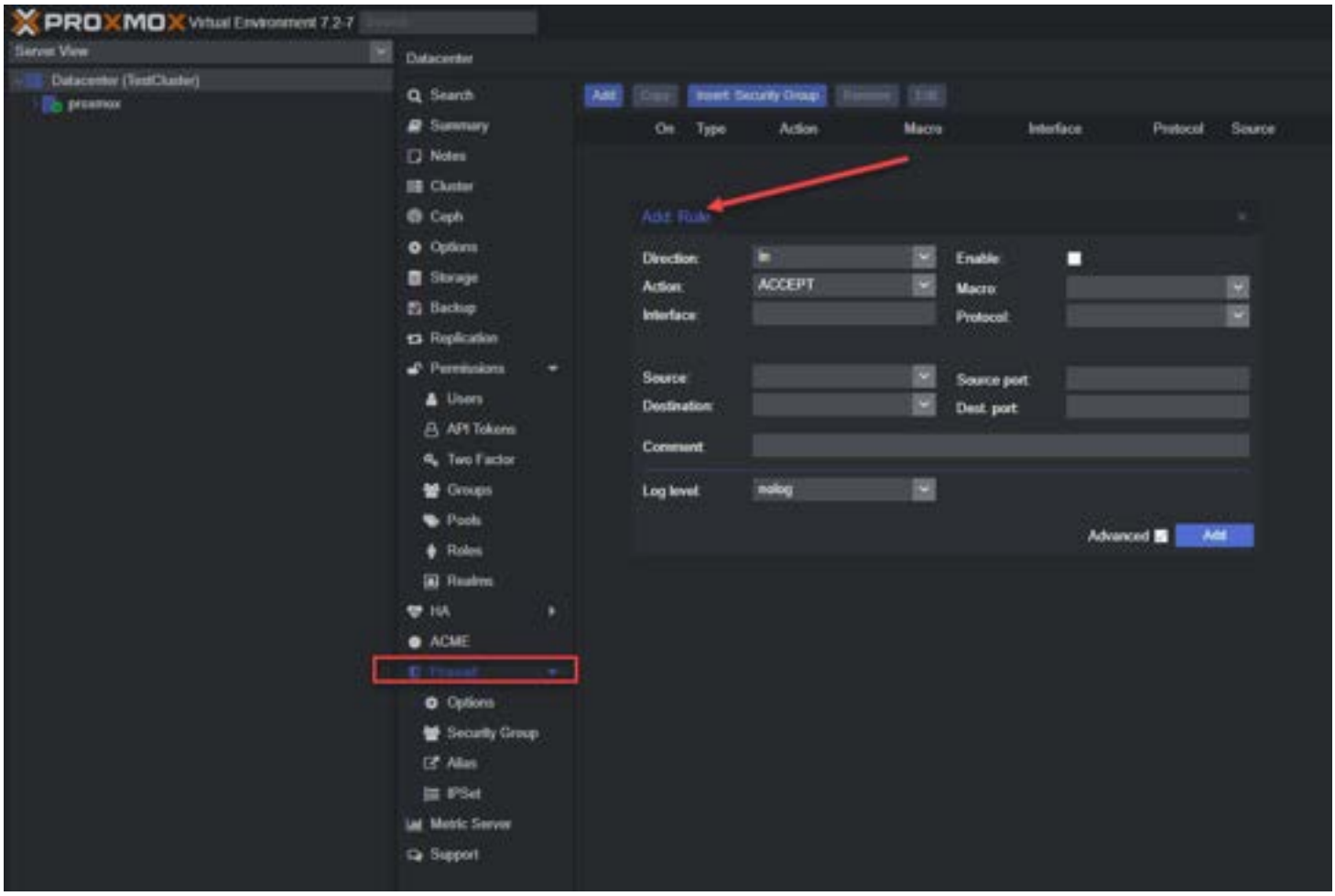
This will start the firewall service and load the firewall configuration files.

Ports used by Proxmox

Proxmox uses several ports for different services, such as SSH, HTTP, and VNC. By default, these ports are open, but you can create firewall rules to restrict access to these ports.

Firewall Rules Configuration Direction

You need to specify the direction of the traffic you want to filter with the Proxmox firewall configuration. You can choose to filter incoming traffic, outgoing traffic, or both.



Enable the Firewall Service from the Command Line

You can enable the Proxmox firewall service from the command line with the following command:

systemctl enable pve-firewall.service

This will start the firewall service on boot and load the firewall configuration files.

Proxmox VE Firewall Configurations via Files

You can also configure the Proxmox firewall using configuration files. These files are located in the **/etc/pve/firewall** directory and define firewall macros, security groups, IP aliases, and firewall rules.

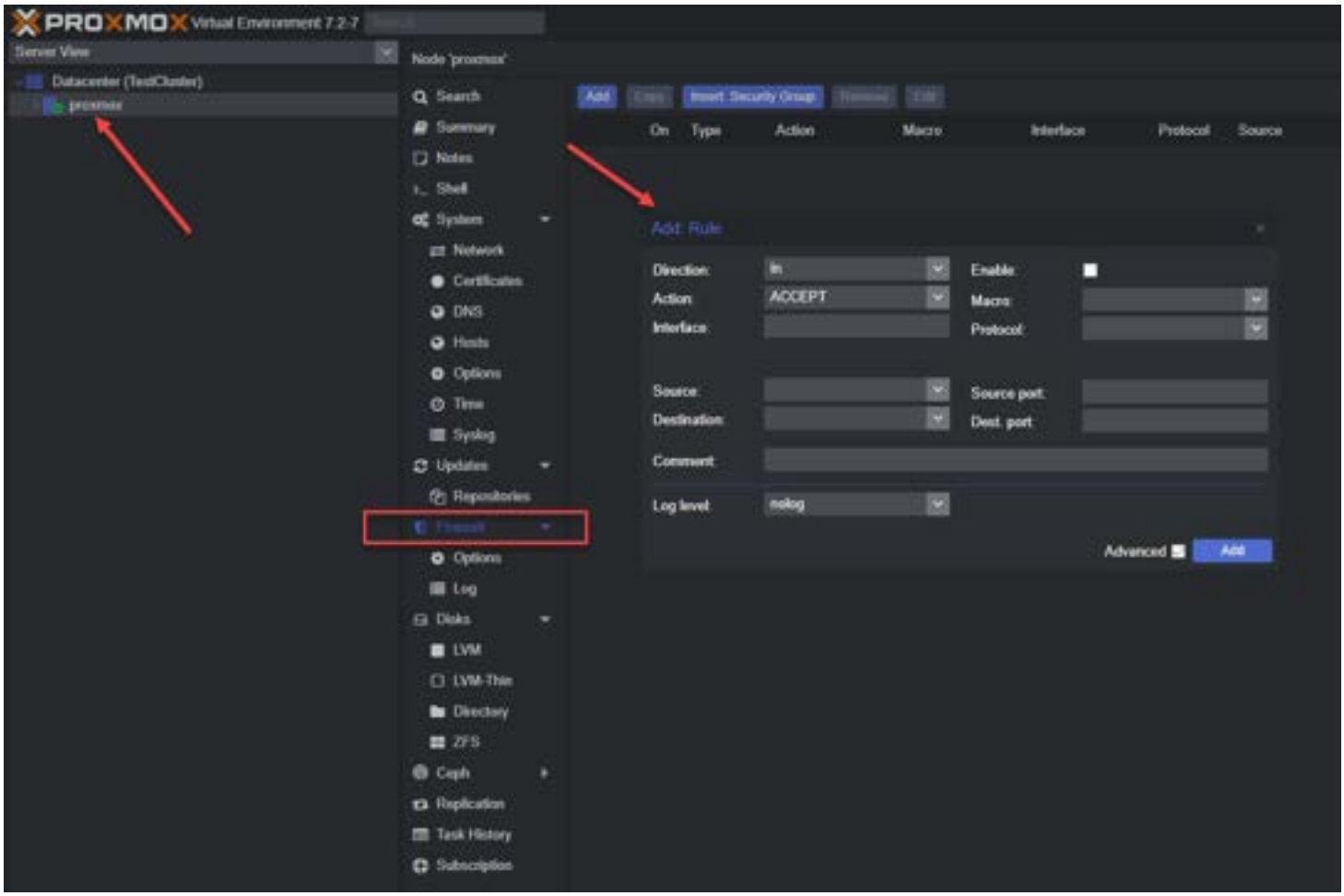
Enabling the Firewall for VMs and Containers

The firewall is disabled by default for virtual machines and containers. However, you can enable the firewall service for a VM or container. To do this, you must add a firewall configuration file to that VM or container's**/etc/pve/firewall** directory.

Host, VMs, and Containers Configuration Files

You can configure the Proxmox firewall using host, VMs, and container configuration files. These files define firewall rules for the respective entities and are located in the **/etc/pve/firewall** directory.

Below is a look at host-specific firewall rules.



See the Generated IPtables Rules

To view the underlying iptables rules generated by the Proxmox firewall:

iptables -L

This will display the current iptables rules managed by the Proxmox firewall service.

Check VM Network Device

When you configure a Proxmox firewall rule for a virtual machine, you need to know the name of the virtual network device. You can find this by using the following command:

qm config <VM ID> | grep net

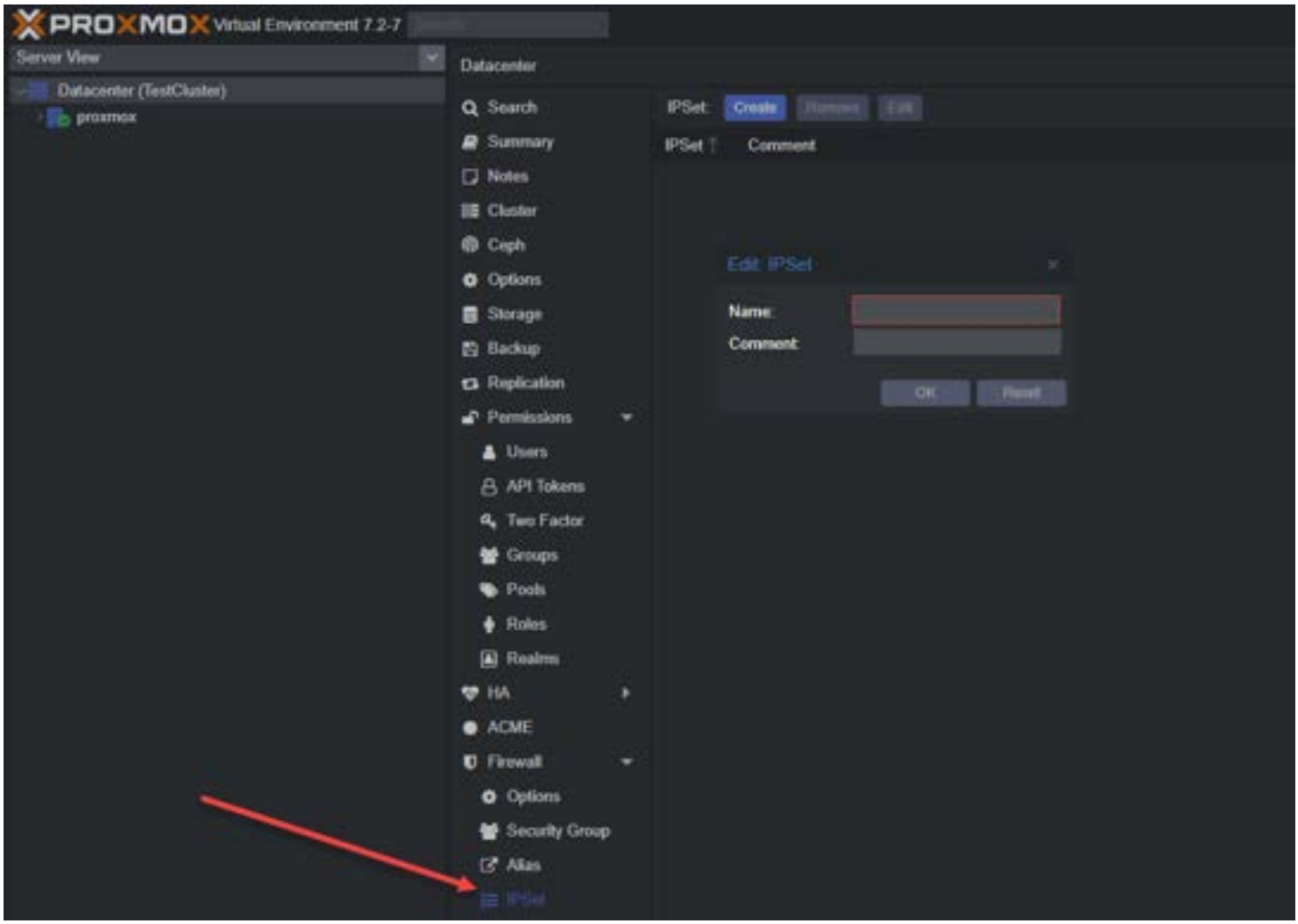
This will display the name of the virtual network device used by the VM.

IP Aliases

You can associate a single IP address with multiple network interfaces with IP Aliases. You can configure IP aliases in the Proxmox firewall using the IP alias configuration file in the **/etc/pve/firewall** directory.

IP Sets

IP sets define a set of IP addresses that can be used in firewall rules. You can configure IP sets in the Proxmox firewall using the IP set configuration file in the **/etc/pve/firewall** directory.



Default Firewall Rules

A set of default firewall rules out of the box allows incoming and outgoing traffic for certain services. These include traffic types such as SSH and HTTP. You can view the default firewall rules using the following command:

iptables -L

```
root@proxmox:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@proxmox:~#
```

Standard IP Alias local_network

An example of the default IP alias is the local_network standard IP alias defined in the Proxmox firewall configuration files. It represents the IP addresses assigned to the Proxmox host and is used in firewall rules to allow traffic between the host and virtual machines/containers.

Adding Security Groups and Rules

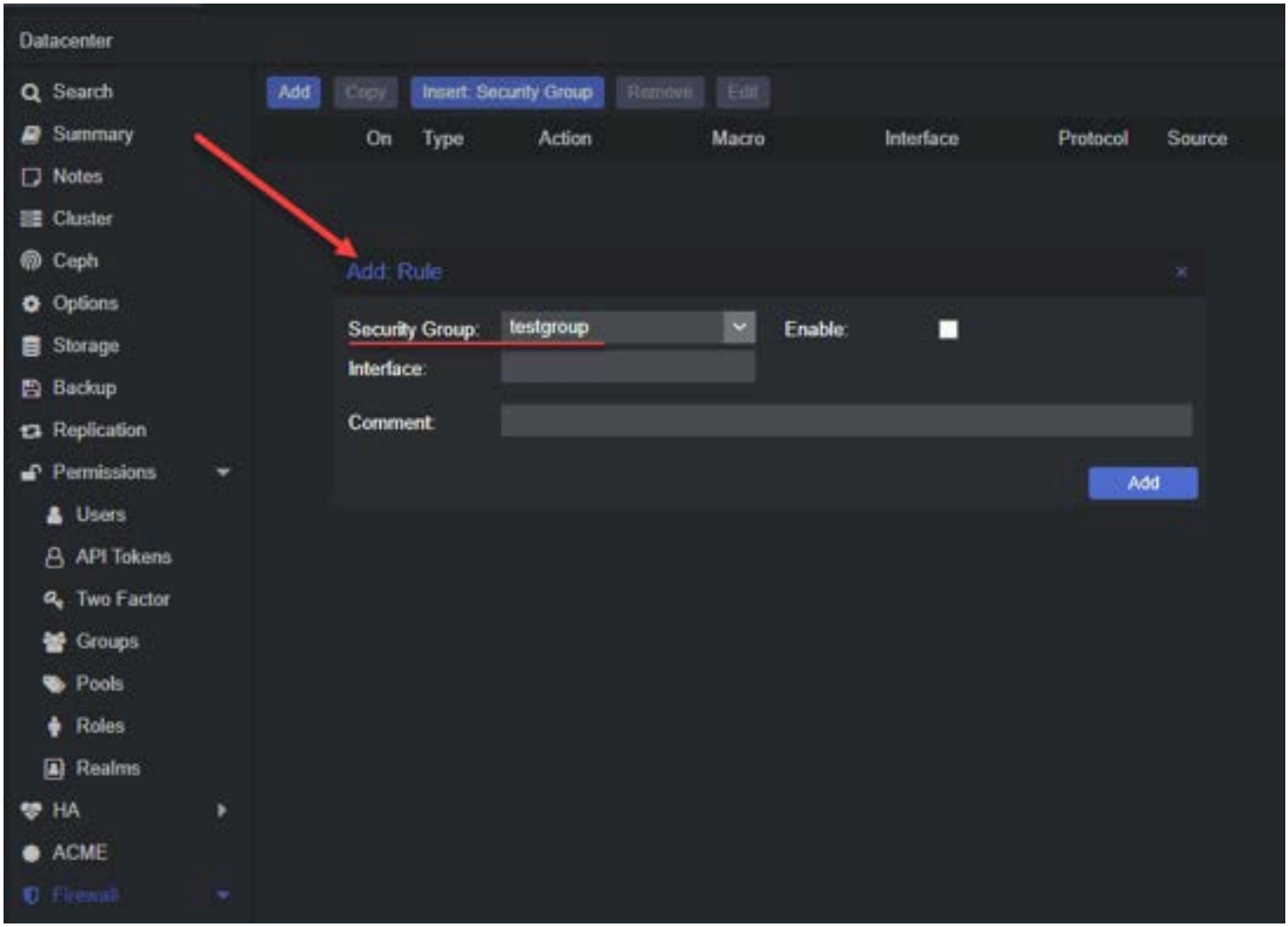
Security groups and IP aliases can be used in the Proxmox firewall configuration files. These can then be used in firewall rules to allow or block traffic based on the group or alias.

Note the following to define security groups and IP aliases using the following syntax in the configuration files:

group <group name> { <ip addresses> }

alias <alias name> { <ip addresses> }

Below is a look at creating security group-based firewall rules.



Supports Both IPv4 and IPv6

The Proxmox firewall supports both IPv4 and IPv6 addresses. You can define rules for both addresses using the same firewall configuration files.

Proxmox Firewall CLI Commands to Know

Several CLI commands are useful when configuring the Proxmox firewall:

1. **pve-firewall enable** – Enables the firewall service
2. **pve-firewall disable** – Disables the firewall service
3. **pve-firewall status** – Displays the current status of the firewall service

4. **pve-firewall reload** – Reloads the firewall configuration files
5. **pve-firewall log** – Displays the firewall log

Adding a Proxmox Firewall Rule

To add a firewall rule to the Proxmox firewall, you must edit the appropriate configuration file in the **/etc/pve/firewall** directory. The syntax for adding a firewall rule is as follows:

iptables -A <zone> -p <protocol> –dport <port> -s <source address> -d <destination address> -j <action>

Cluster Nodes

If you are using a Proxmox cluster, you can configure the firewall rules to apply to all nodes in the cluster. This is done by configuring the underlying iptables rules automatically and using the same firewall configuration files on all nodes.

PVE Firewall Status

Note the following to check the status of the Proxmox PVE firewall service using the **pve-firewall status** command on the CLI.

```
USAGE: pve-firewall <COMMAND> [ARGS] [OPTIONS]

       pve-firewall help [<extra-args>] [OPTIONS]

       pve-firewall compile
       pve-firewall localnet
       pve-firewall restart
       pve-firewall simulate  [OPTIONS]
       pve-firewall start  [OPTIONS]
       pve-firewall status
       pve-firewall stop
root@proxmox:~#
```

Define Rules

To define a firewall rule in the Proxmox firewall, you need to edit the appropriate configuration file in the **/etc/pve/firewall** directory. The syntax for adding a firewall rule is as follows:

iptables -A <zone> -p <protocol> –dport <port> -s <source address> -d <destination address> -j <action>

Outgoing Traffic

You can also configure the Proxmox firewall to filter outgoing traffic based on the destination IP address, protocol, and port.

Required Firewall Rules

Certain firewall rules are required for the Proxmox VE software to function properly. These rules allow traffic for SSH, HTTP, and VNC.

Generated Iptables Rules

The underlying iptables rules generated by the Proxmox firewall can be viewed using the **iptables -L** command.

Automatically Distributed

If you are using a Proxmox cluster, the underlying iptables rules for the firewall are automatically distributed to all nodes in the cluster.

PVE Firewall Stop

To stop the Proxmox firewall service use the **pve-firewall stop** command on the CLI.

CLI Commands

Several CLI commands can be used to manage the Proxmox firewall service, such as **pve-firewall enable**, **pve-firewall disable**, **pve-firewall status**, and **pve-firewall reload**.

Remote IPs

You can manage access from remote IP addresses to bolster security. You can configure firewall rules for remote IPs using the **remote.fw** file located in the **/etc/pve/firewall** directory.

Cluster Specific Firewall

Using a Proxmox cluster, you can configure a cluster-specific firewall using the **/etc/pve/firewall/cluster.fw** configuration file.

Configuration Files

The Proxmox firewall is configured using several configuration files in the **/etc/pve/firewall** directory. These files define firewall macros, security groups, IP aliases, and firewall rules.

Corosync Cluster Traffic

You can manage traffic for Corosync cluster traffic. You can configure firewall rules for Corosync cluster traffic using the **corosync.fw** file located in the **/etc/pve/firewall** directory.

HTTP Traffic

You can also filter HTTP traffic in your Proxmox environment. You can configure firewall rules for HTTP connections using the **http.fw** file located in the **/etc/pve/firewall** directory.

Create Rules

When creating a firewall rule you need to edit the appropriate configuration file in the **/etc/pve/firewall** directory. The syntax for creating a firewall rule is as follows:

iptables -A <zone> -p <protocol> –dport <port> -s <source address> -d <destination address> -j <action>

Wrapping up

The Proxmox firewall is a great tool admins can use to manage and control traffic to the Proxmox data center, Proxmox hosts, and virtual machines and containers running in the environment. The firewall is based on the Linux iptables firewall and is managed using several configuration files located in the **/etc/pve/firewall** directory.

The Proxmox cluster firewall rules are distributed in nature and synchronized between all cluster nodes. It is a great capability that can effectively help secure workloads and Proxmox environments.

Proxmox Container vs VM features and configuration

September 29, 2022
Proxmox

d0dcd259 26f5 482f bda7 b08b4ef21489

Proxmox is an extremely versatile hypervisor that provides the ability to run a Proxmox Container vs VM as part of the native functionality built into the platform. This is a great way to have the best of both worlds. It allows for solving multiple applications challenges and multiple purposes with a single hypervisor. You can use these for production or test environments.

Instead of running multiple virtual machines (VM workloads) to host services, you can run the LXC containers on the host system for more efficient environments. Let's explore the topic of Proxmox containers vs VM instances and see how running virtual machines in Proxmox differs from Proxmox containers.

Take note of my latest Promox posts:

- Proxmox Management Interface VLAN tagging configuration
- Proxmox vs ESXi – ultimate comparison 2022
- Proxmox Create ISO Storage Location – disk space error
- pfSense Proxmox Install Process and Configuration
- Proxmox Update No Subscription Repository Configuration

What is the difference between LXC containers and Docker containers?

First of all, many will recognize that we are describing Proxmox VE containers as LXC containers and not Docker. What is the difference between the two? LXC containers are known as Linux Containers and are an OS-level virtualization technology.

It enables running multiple Linux OS'es on a single LXC host. LXC containers are much smaller than a full virtual machine but often larger than Docker containers.

This can help with the performance of spinning up applications and setup access much more quickly to resources. There are many reasons why one is preferred over the other. However, depending on the use case, one may be the best choice over the other.

Docker is most popular

Docker containers are arguably the most popular container technology used in the enterprise today. They focus on running applications and all their dependencies in a seamless, self-contained way and allow provisioning a single-purpose application environment for running applications.

LXC containers are more like virtual machines

LXC containers are very much like a virtual machine, but significantly lighter weight since it is sharing the host kernel with the LXC host. It does not require the disk space or other resources as full VMs.

LXC containers aim to align with a specific distribution of Linux. However, Docker containers aim to be distro-less and focus on the applications and dependencies. Virtual machines have their own kernel instance as opposed to the shared kernel instance with containers.

Allow hosting multiple applications

With multiple Docker containers, you can host multiple applications on your container host. LXC containers provide the traditional resources you would find in a virtual machine running in the same environment. However, you can't run different operating systems like Windows in an LXC container, only different Linux distributions.

Nesting Docker containers inside LXC containers

One of the really cool things about running LXC containers on a Proxmox host is you can actually install Docker inside an LXC container. In fact, you can run Kubernetes in a lab environment using LXC containers as your Kubernetes hosts.

Many may not realize that Docker is actually a fork of Linux LXC containers. Both LXC and Docker share the same kernel with the container host.

Proxmox container vs virtual machine

Proxmox, unlike many commercial hypervisors, has the ability out of the box to run containers on top of the hypervisor directly. You can choose to create either a container vs VM.

Container vs VM

A virtual machine can load any operating system you want inside the VM with its kernel instance and provides the best isolation for running a server for resources. Containers share the kernel instance with the physical server Linux instance.

So the container operating system is shared with the host. Both have the hardware abstracted using virtualization technologies. The user does not know they are accessing virtual machines or containers when accessing resources.

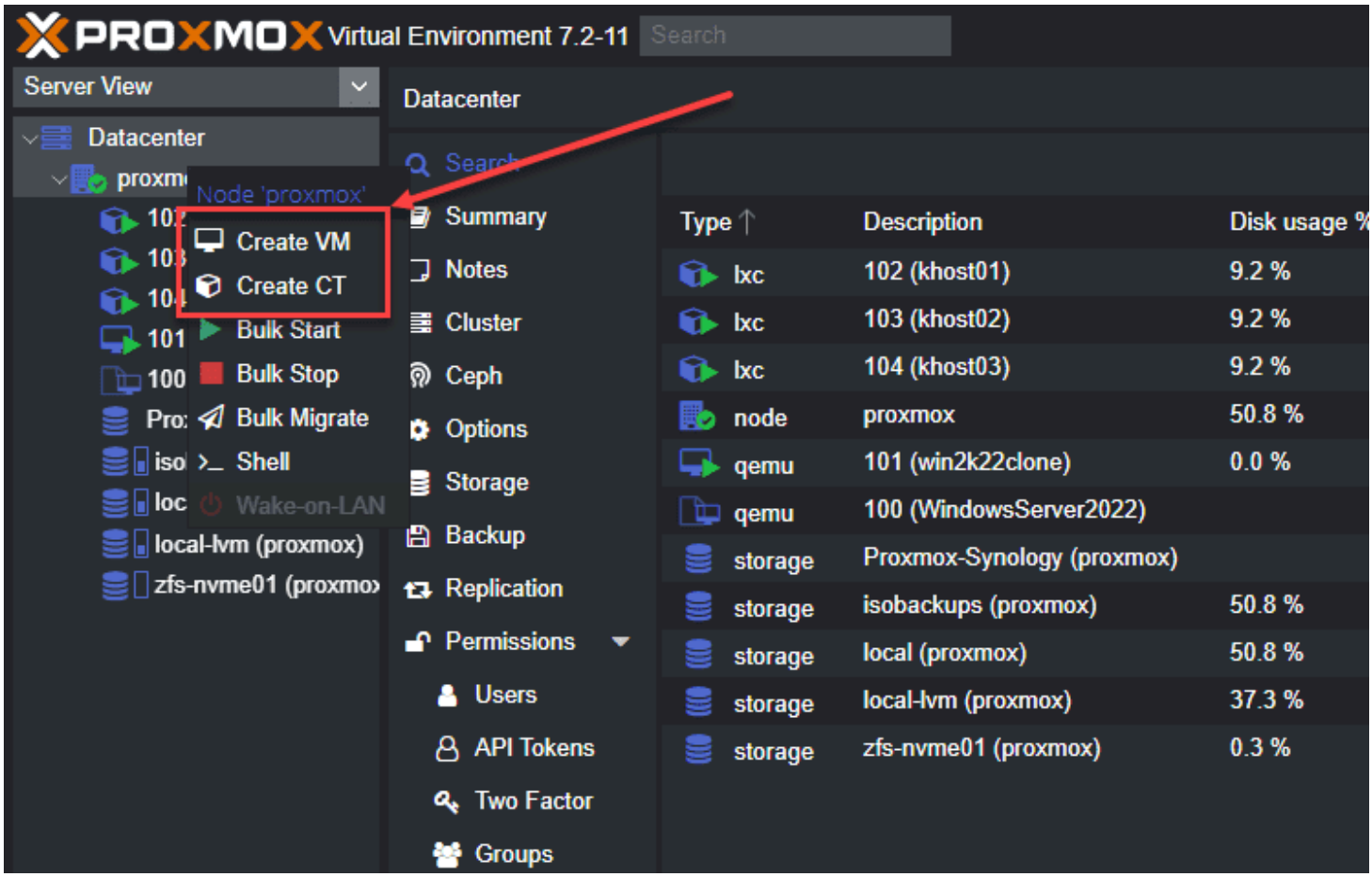
Overhead

The overhead of running multiple virtual machines is much more than the overhead of running multiple containers. If users need access to a desktop or desktop resources, virtual machines are needed for this purpose. The speed to provision containers is faster, and the effort involved is generally less involved.

Persistence

Virtual machines are generally considered persistent and have to maintain lifecycle management, etc. Whereas containers offer the ability to have ephemeral resources. The time to boot a container is minimal.

You will see the choice in the menu for Create VM or Create CT on the host system. Again the main difference is you are creating a full virtual machine or an LXC container.



Backups

In terms of backups, you can backup both containers vs VM in Proxmox VE. This is a great option since many solutions allow backing up virtual machines but do not support containers.

Creating new Proxmox containers

Let's look at the configuration steps to create Proxmox containers and see what configuration is involved. Incidentally, the screens for creating a virtual machine are basically the same. so, we will look at the containers screens since these are probably the least familiar Again, with containers, we are using a virtualization option that shares the same kernel instance with the Proxmox host.

When you choose the New CT option, you will begin the **Create: LXC Container** wizard. Below you will see the first screen has you define:

Node

CTID

Hostname

Privileges

Nesting

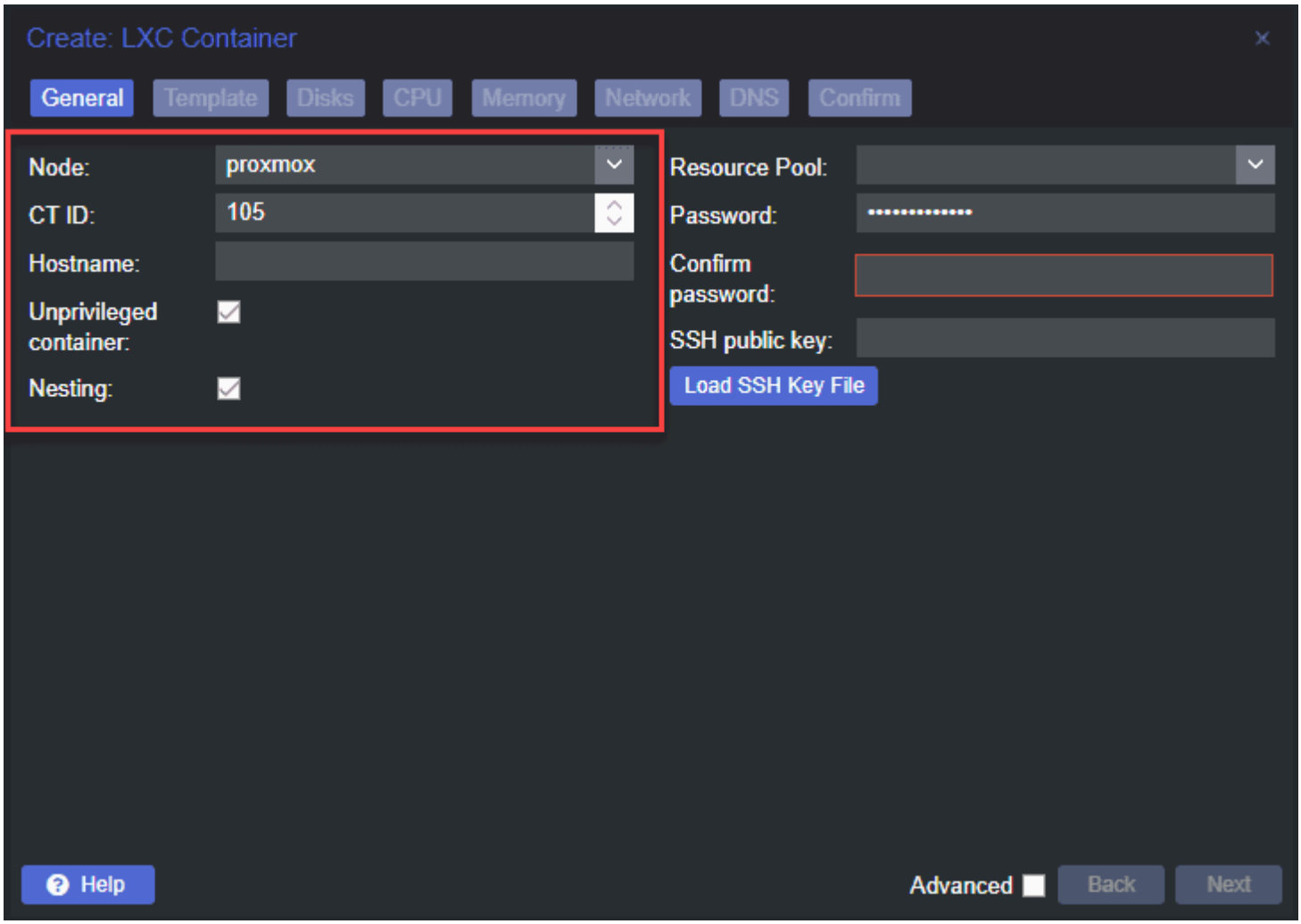
Resource Pool

Password

SSH public key

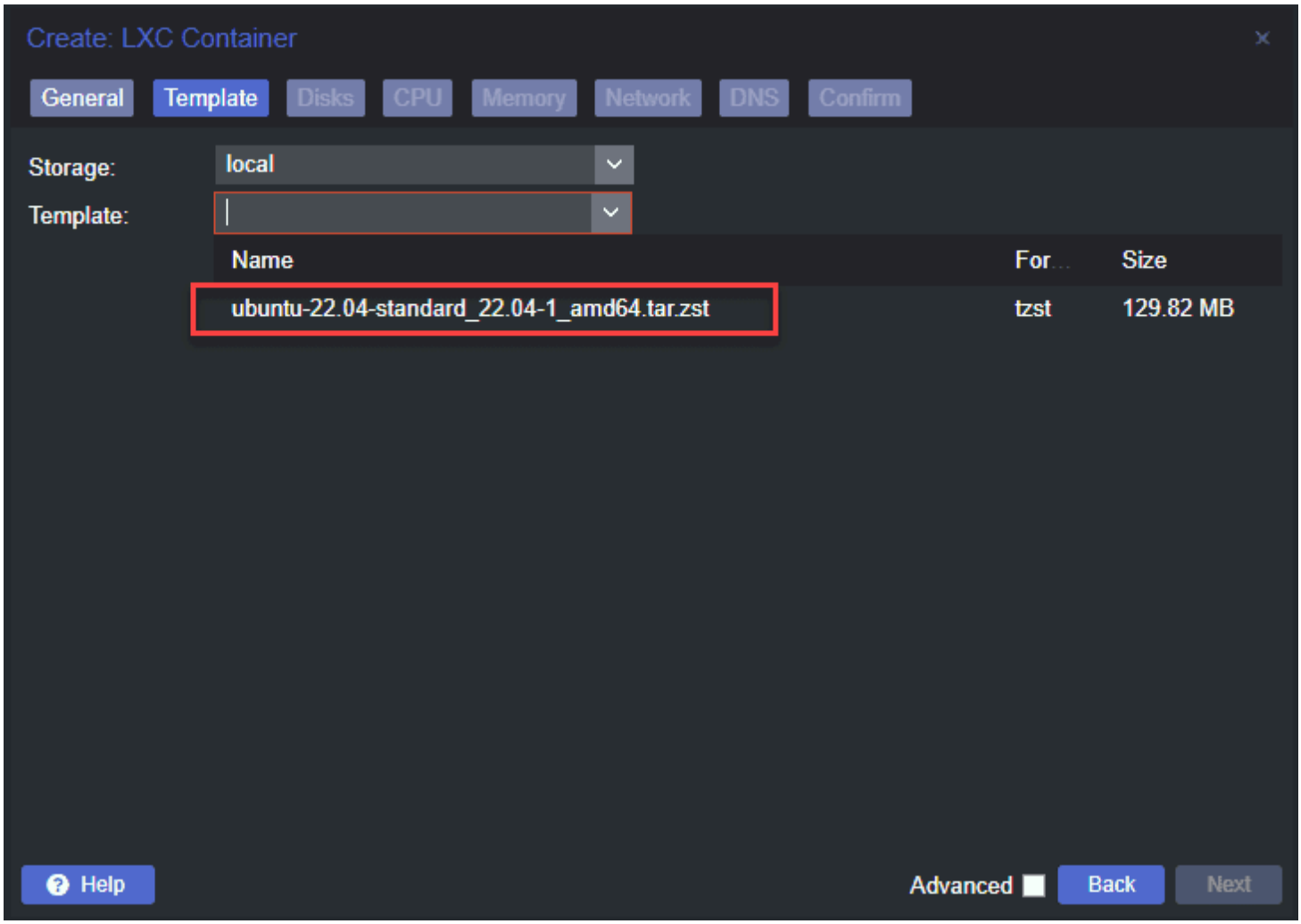
General Tab

This screen helps establish the basics of connectivity, authentication, and a few other data configurations for the container instance.



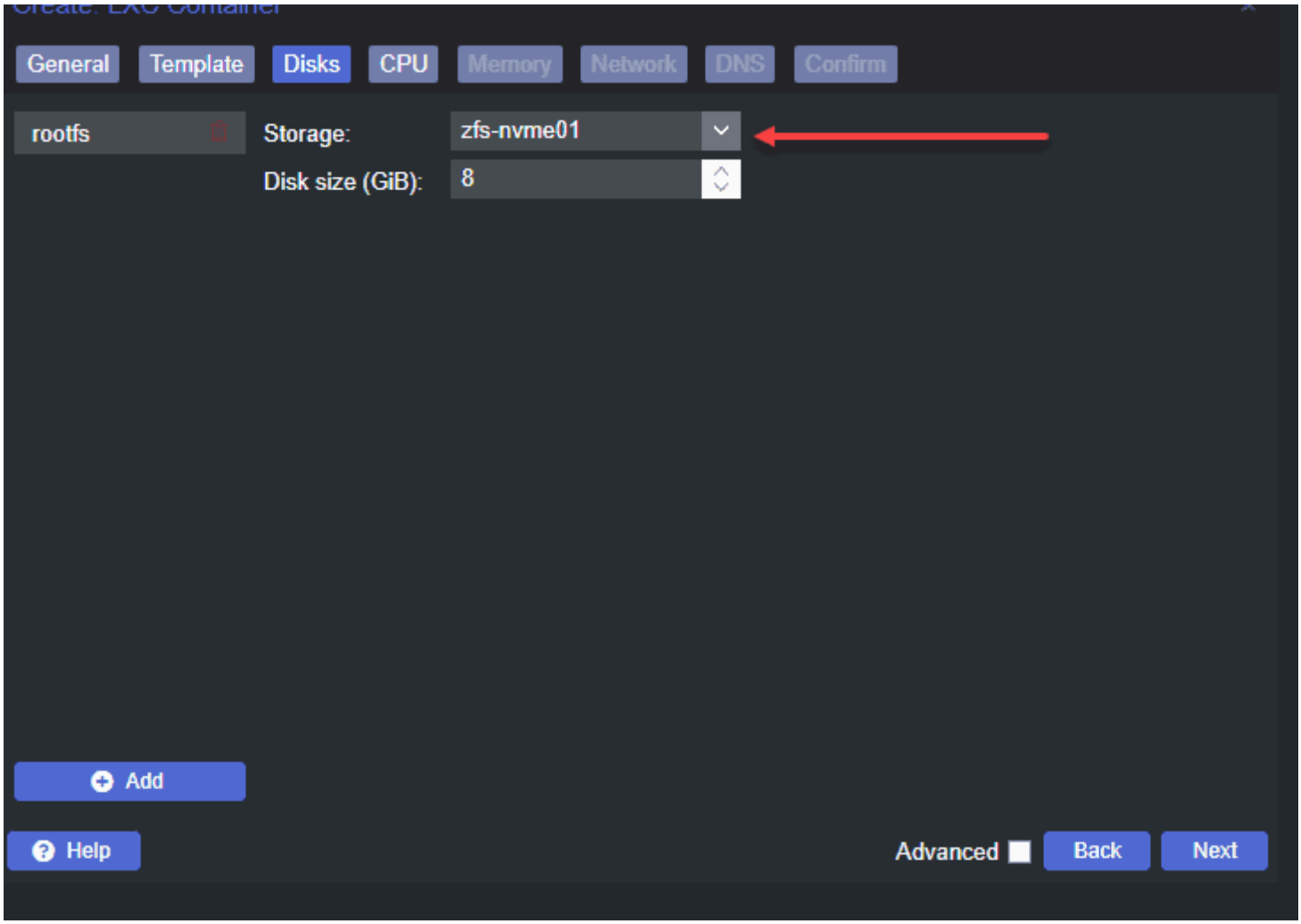
Choosing your container template

On the next screen, you choose the Proxmox containers template that will be used for spinning up the LXC container. As you can see below, I have pulled down an Ubuntu 22.04 container image to spin up a new system.



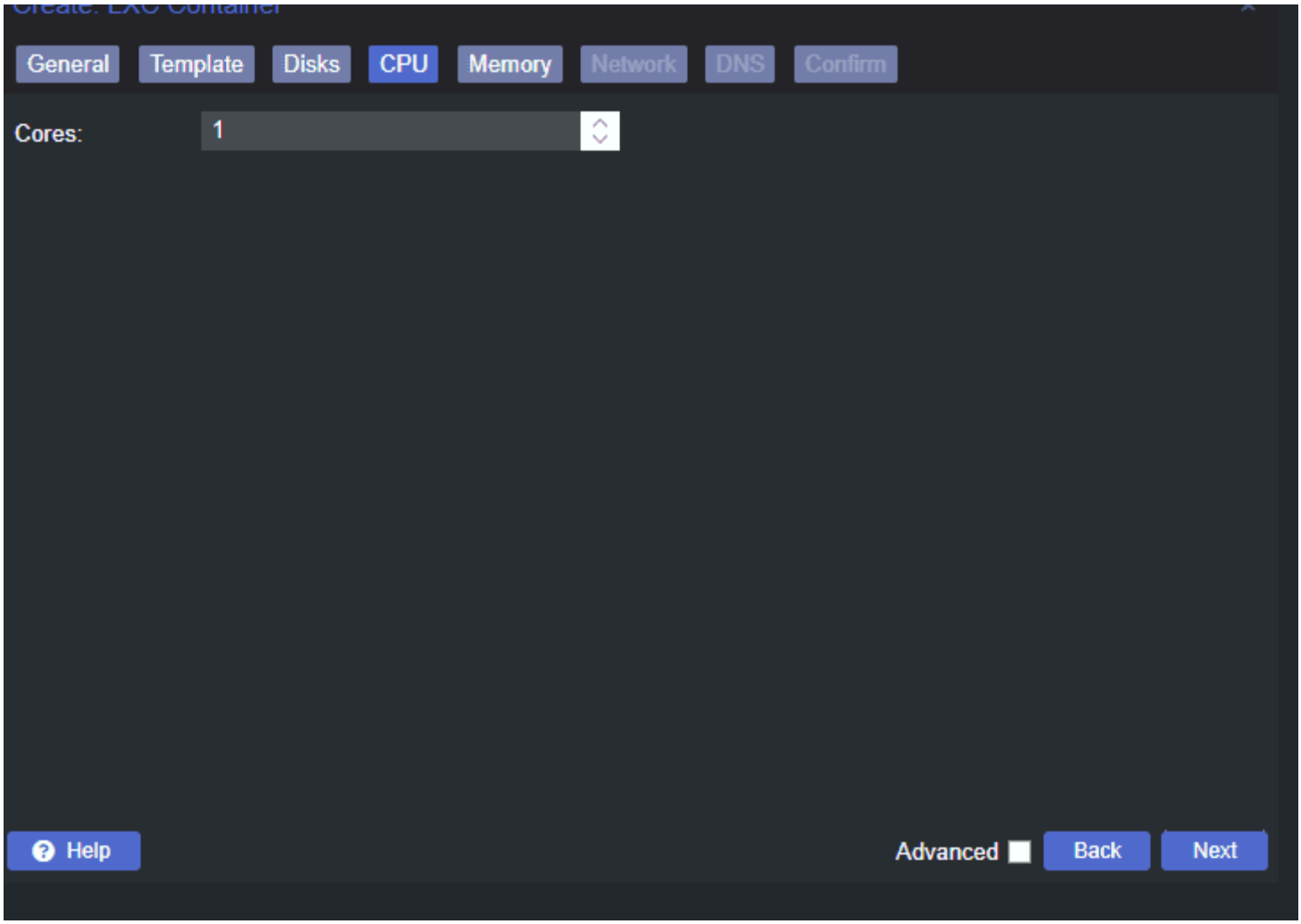
Choosing storage

Next, we select the disk storage needed for the LXC container. Below, I have selected the storage for the container file storage using the Proxmox tool.



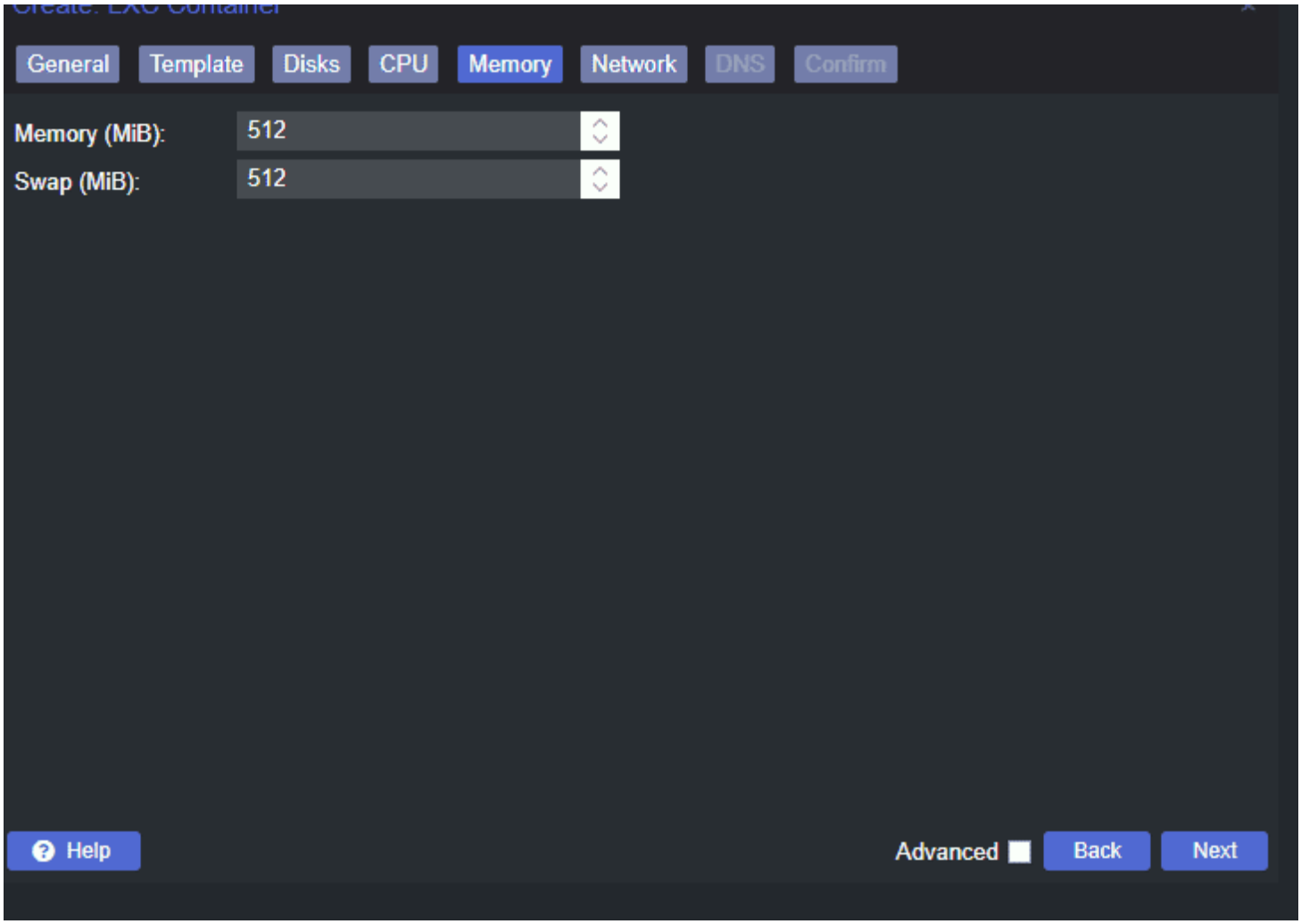
Configuring the CPU settings

Next, we select the CPU resources, needed for the container. We can select the core value needed for the new container.



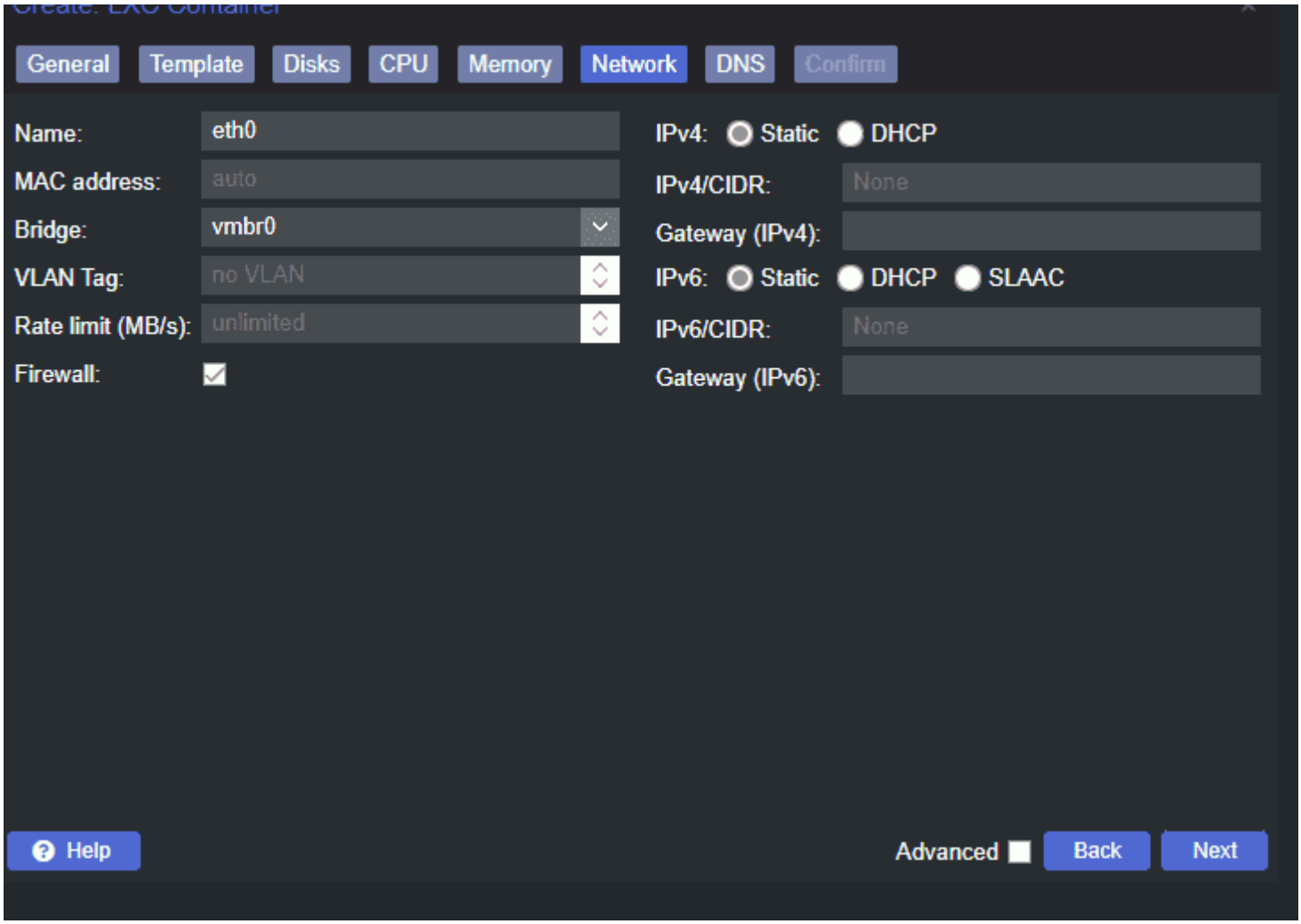
Configuring memory

We need to assign the memory value for the new container in Proxmox.



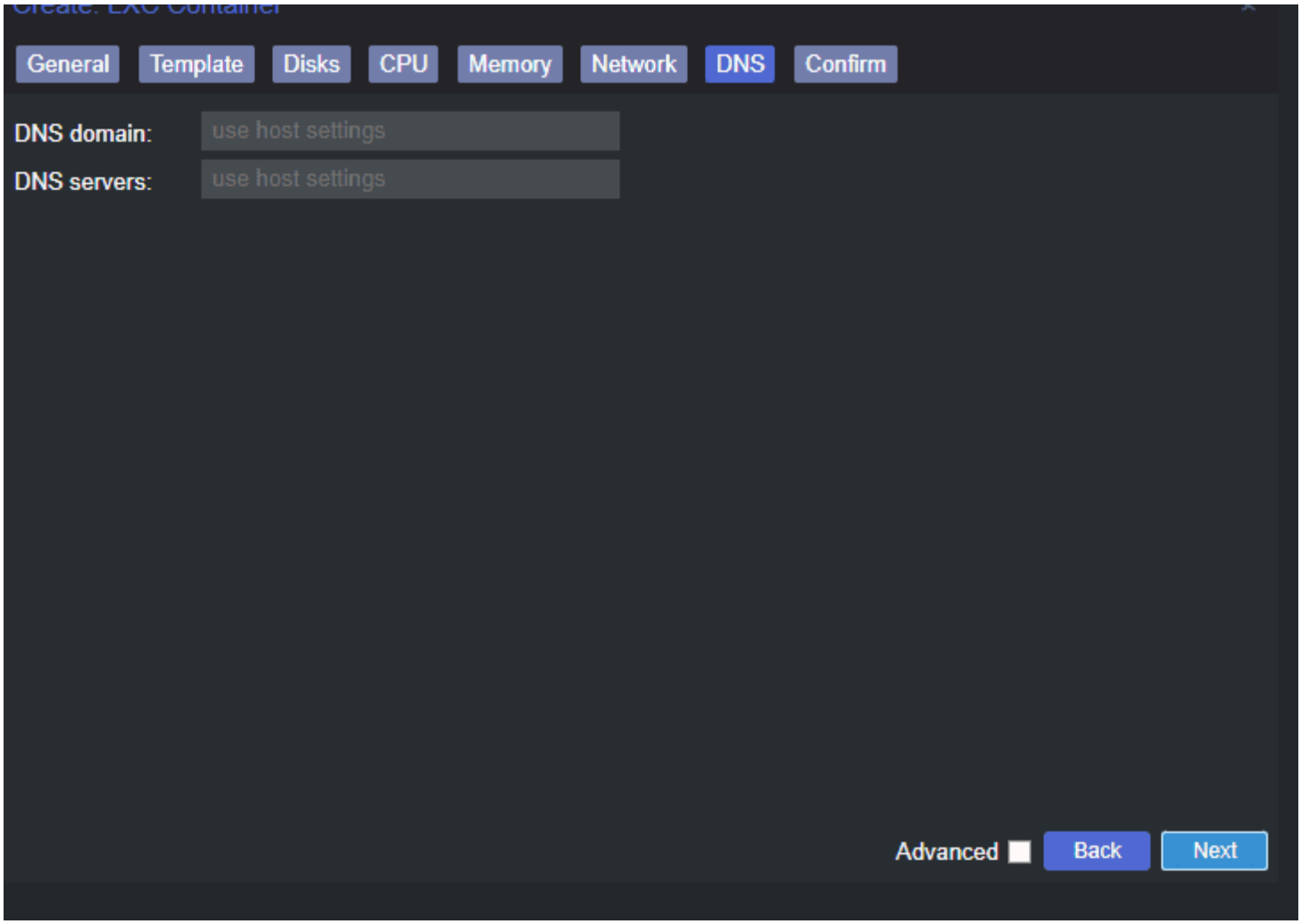
Network configuration

Now, we create network resources for the new LXC container running in Proxmox. The Proxmox containers can have all of the normal virtual machines configuration we are used to, such as assigning a VLAN tag, IP address configuration, such as static or DHCP and others as you would any other computer system running on Proxmox VE.



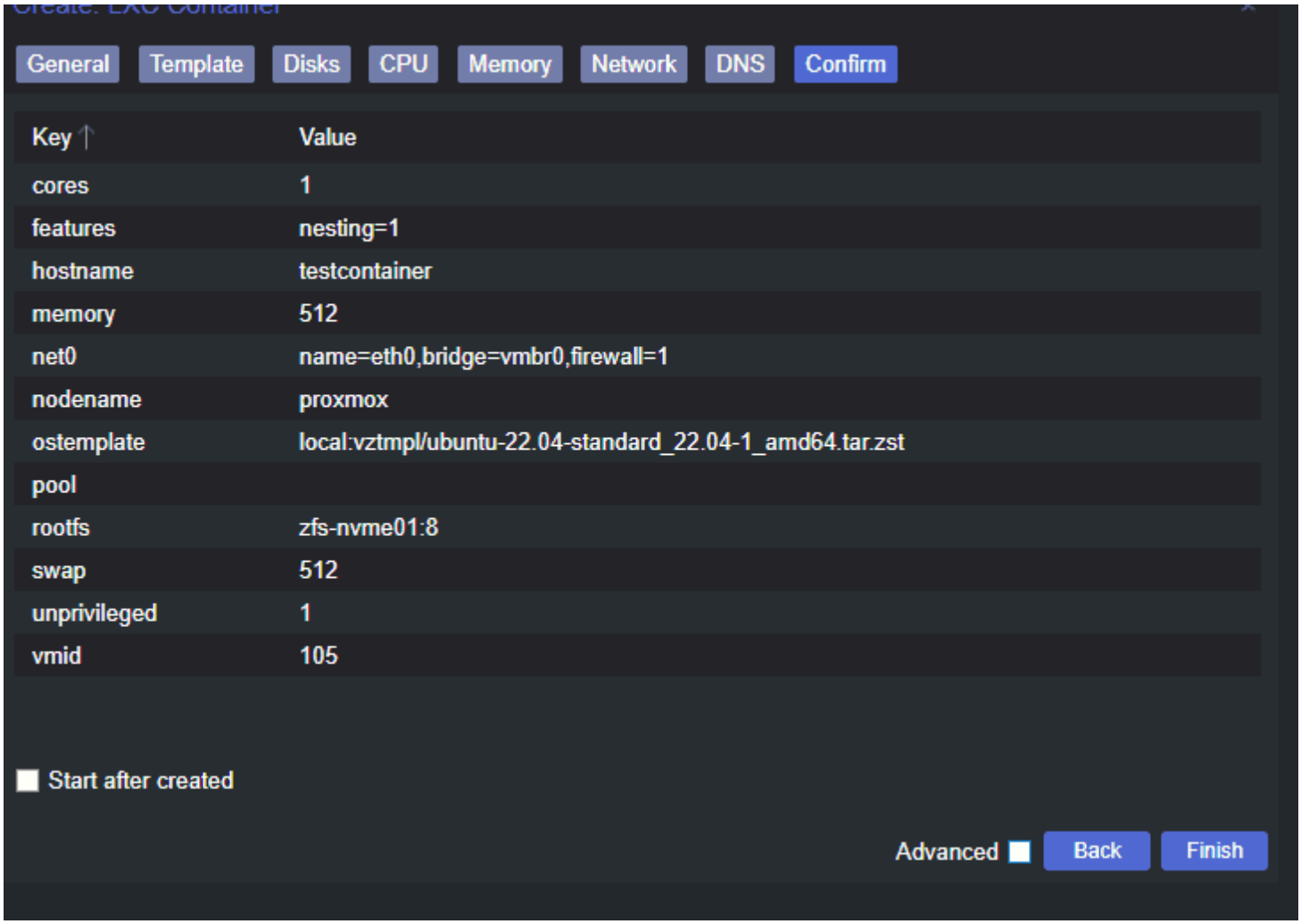
DNS configuration

Going along with the network configuration on the next screen we have the DNS configuration.



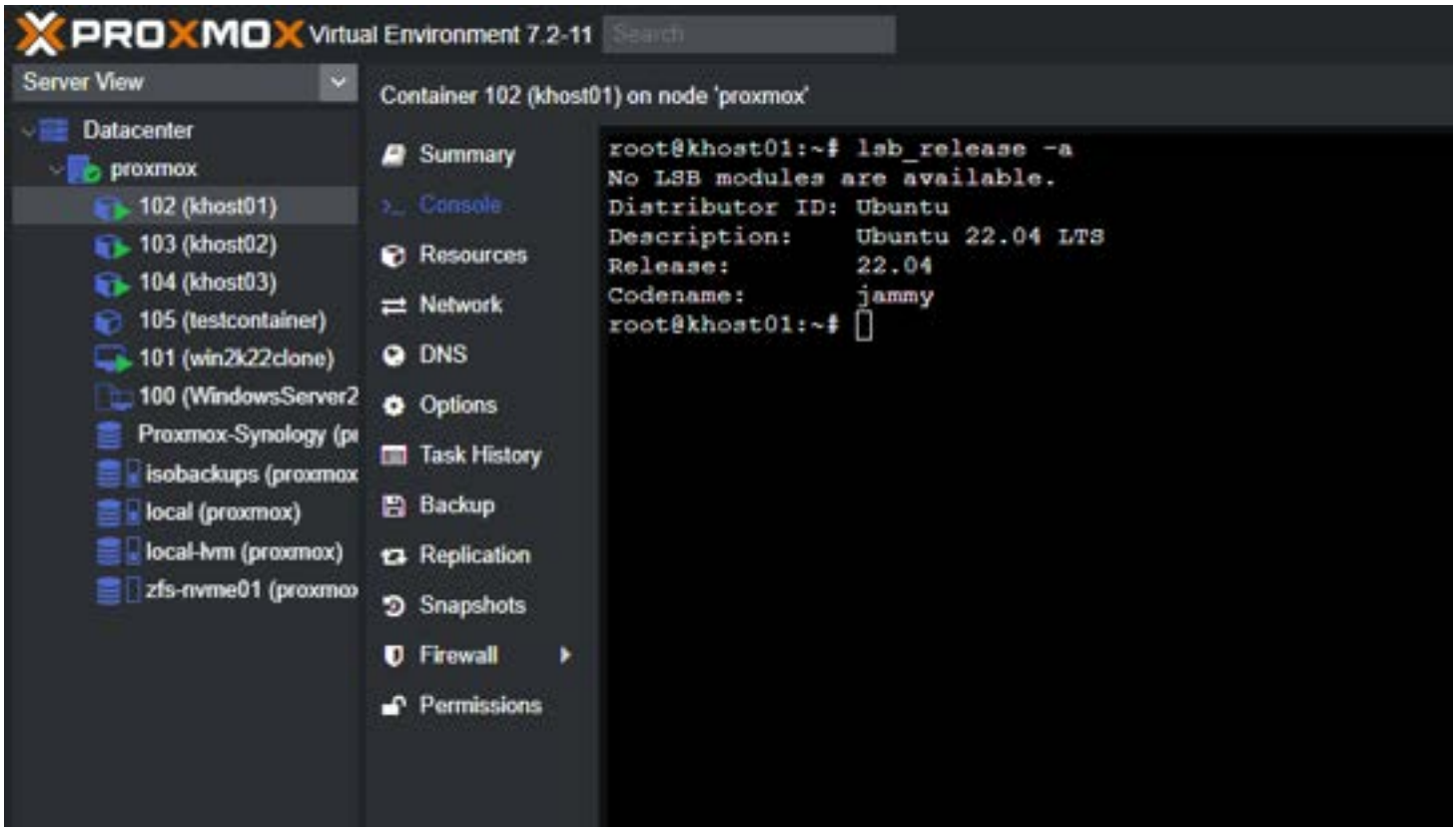
Confirming the creation of the new LXC container

Finally, we get to the point of finishing out the Proxmox VE configuration. Here we can review the



Accessing the console of the container for command line access

Below, you can easily access the container's command line from the Proxmox VE web interface.



Converting virtual machines and containers to templates

In Proxmox VE, you can convert both virtual machines and containers to templates. Templates are a way to easily save a copy with the configuration included for a virtual machine or a container so these can be quickly spun up from the template.

You can have Windows, Linux, and other operating systems converted to template and easily spin these up for quick deployment from a common mount point.

Proxmox container vs VM FAQs

What are Proxmox containers? Proxmox containers are LXC containers that are very similar to virtual machines in terms of features and behaviors. These are heavier containers generally speaking than Docker containers. Docker containers focus on applications, whereas LXC containers focus on Linux distributions.

What are Promox containers vs VM? Containers vs VM in Proxmox VE provides very robust and diverse capabilities that allow solving many different challenges from a technical and business perspective.

What is the different between Docker vs. LXC containers? Docker is focused on applications and LXC containers are focused on distributions and more VM-specific functionality.

Wrapping Up

Promox has a wide range of features. When looking at Proxmox container vs VM functionality, it covers it all. Using LXC containers you can quickly spin up environments. Virtual Machines allow spinning up isolated environments with their own kernel instance for the most isolation. However, containers are still a secure way to run applications and spin up environments for users to access applications and resources.

Proxmox Containers with Fedora CoreOS Install

February 14, 2024
Proxmox

Proxmox containers with fedora coreos

I recently looked at the installation of Fedora CoreOS on VMware. In the home lab, many are running Proxmox and maybe more coming up will be switching from VMware over the course of 2024. Let's take a look at the topic of running Proxmox Containers with Fedora CoreOS setup.

Table of contents

Proxmox and Fedora CoreOS

Combining an excellent hypervisor for virtualization and an operating system platform that is **purpose-built for containerization and Kubernetes**, is a great combination. We can do this by running Fedora CoreOS distribution on top of Proxmox VE with KVM for the purpose of running containers.

Fedora CoreOS

While you can run LXC container (Linux containers) configurations and container template machines inside of Proxmox natively, many developers and DevOps guys need access to Docker containers. In the home lab, most solutions you want to self-host are readily available as Docker containers also without running full virtual machine instances with full operating systems. So, running Docker is a great way to have access to these solutions.

Fedora CoreOS has as its focus, containerized infrastructure. If you look at the documentation, It offers an automatically updating, minimal, container-centric operating system that natively runs Docker, Podman, and can run Kubernetes as well also, with good support across the board. It is also **immutable which means it has security enhancements** for customers running it for their container cluster.

So if you are looking for a clean, secure, and immutable OS to run your containers on top of Proxmox, CoreOS is a great solution!

Proxmox

Running it on top of Proxmox has other benefits such as the ability to run Proxmox Backup Server solution to backup the host virtual machines. It has powerful networking and monitoring. Businesses can even choose to have enterprise support and home labbers can become a member of the Proxmox support forum (for search forums threads, requests, and share things with others in the home forums such as issues and troubleshooting) and access to other Proxmox solutions, like Proxmox Mail Gateway.

Fedora CoreOS install on Proxmox

Let's look at the Fedora CoreOS installation on Proxmox VE and see what steps are involved. There is actually a really great community project that will allow getting up and running with Fedora CoreOS on Proxmox. We will take a look at this below. In addition, Fedora CoreOS can be installed on bare metal using a live ISO installation as well as OVA appliance for VMware.

Note the steps we will cover:

1. Clone the community repo
2. Enable snippets on a Proxmox VE storage repository
3. Run the included shell script
4. Configure **cloud-init** options for the created template
5. Clone the template VM to a new Fedora CoreOS virtual machine

1. Clone the community repo

Before creating containers with Fedora CoreOS, ensure your Proxmox VE setup is ready. This involves checking available disk space, configuring network settings, and making sure your Promox host is up-to-date. If you can navigate to the Proxmox web interface to manage containers (view Proxmox container toolkit settings) and virtual machines you should be good to go.

We need to clone the repository that contains the community script for deploying the Fedora CoreOS installation. You can clone the following repository:

- https://github.com/GECO-IT/fedora-coreos-proxmox.git

I did this directly from my Proxmox VE host. Just install the git tools if you haven't already: apt install git -y

You will see the **fedora-coreos-proxmox** folder. If you cd inside the folder, you will see a **vmsetup.sh** wrapper script that is the script we will run on the Proxmox VE server. The **fedora-coreos-<version>.yaml** serves as the ignition file for CoreOS. The Ignition file configures all the required settings..

```
10.1.149.58 - PuTTY
root@pmox01:~# git clone https://github.com/GECO-IT/fedora-coreos-proxmox.git
Cloning into 'fedora-coreos-proxmox'...
remote: Enumerating objects: 31, done.
remote: Counting objects: 100% (31/31), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 31 (delta 13), reused 31 (delta 13), pack-reused 0
Receiving objects: 100% (31/31), 68.80 KiB | 704.00 KiB/s, done.
Resolving deltas: 100% (13/13), done.
root@pmox01:~# ls
fedora-coreos-39.20240112.3.0-qemu.x86_64.qcow2.xz  fedora-coreos-proxmox
root@pmox01:~# cd fedora-coreos-proxmox
root@pmox01:~/fedora-coreos-proxmox# ls
fcos-base-tmplt.yaml  hook-fcos.sh  LICENSE  README.md  screenshot  vmsetup.sh
root@pmox01:~/fedora-coreos-proxmox#
```

Cloning the repo down from the community to install fedora coreos in proxmox

2. Enable snippets on a Proxmox VE storage repository

Next, following the instructions on the GitHub repo, we need to enable **snippets** on a local storage repository. It defaults to the **local** default storage in Proxmox. However, you can edit the **vmsetup.sh** script to point to a different storage location for both the snippet storage and template storage which is also needed.

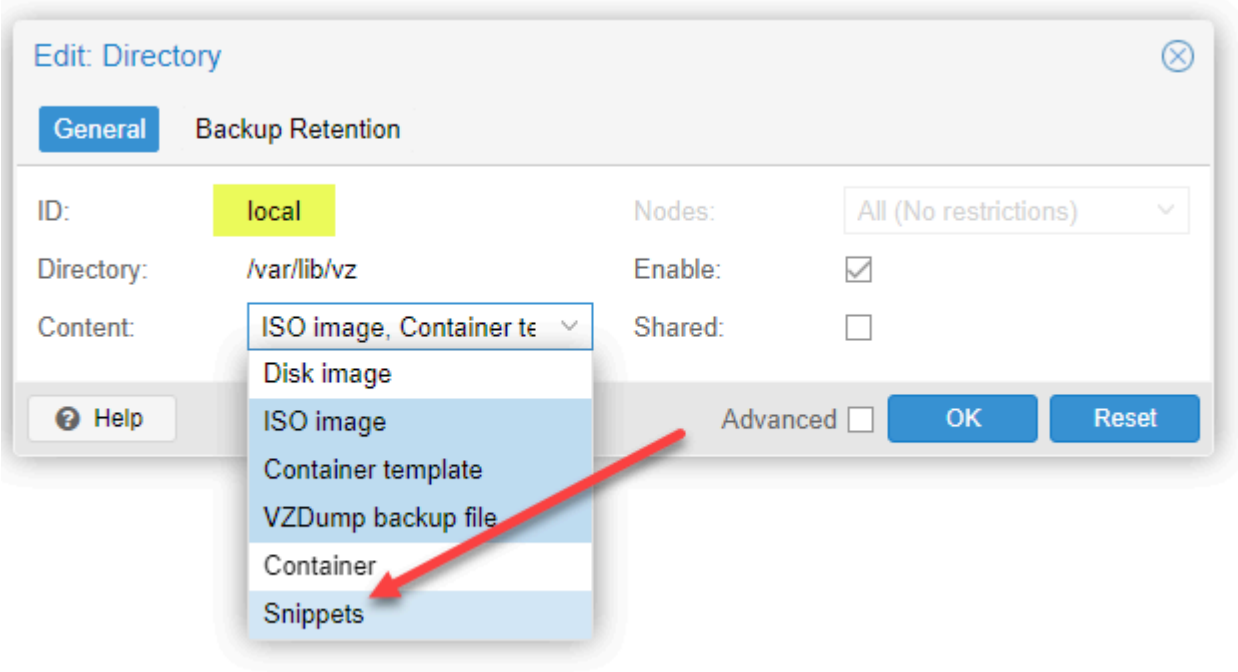
You will see the lines I have changed below in the top part of the script. I have changed the **TEMPLATE_VMSTORAGE** and **SNIPPET_STORAGE** to the locations I wanted. Also, the vmsetup.sh script was quite a bit behind on the Fedora CoreOS version it was looking to deploy. So, I updated that to the latest at the time of writing in the file below.

```
#!/bin/bash

#set -x # debug mode
set -e

# ==========================================================================================
# global vars

# force english messages
export LANG=C
export LC_ALL=C

# template vm vars
TEMPLATE_VMID="900"
TEMPLATE_VMSTORAGE="CephPool01"
SNIPPET_STORAGE="cephfs"
VMDISK_OPTIONS=",discard=on"

TEMPLATE_IGNITION="fcos-base-tmplt.yaml"

# fcos version
STREAMS=stable
VERSION=39.20240112.3.0
PLATEFORM=qemu
BASEURL=https://builds.coreos.fedoraproject.org

# ==========================================================================================
# main()
```

Changing the template and snippet storage in the shell script

Make a connection in a browser to the URL of your Proxmox web UIIf you want to go with the default **local** location, or any other location, navigate to **Datacenter > Storage > "your storage"** in the menu navigation, then click **EDIT**.

Add **Snippets** to the **Content** dropdown. Then click **OK**. You can also create a new folder and enable it with the snippets content type if you want something specific set aside for this use case. Just create a new folder, enter a description if you like, and make changes to the vmsetup.sh file.

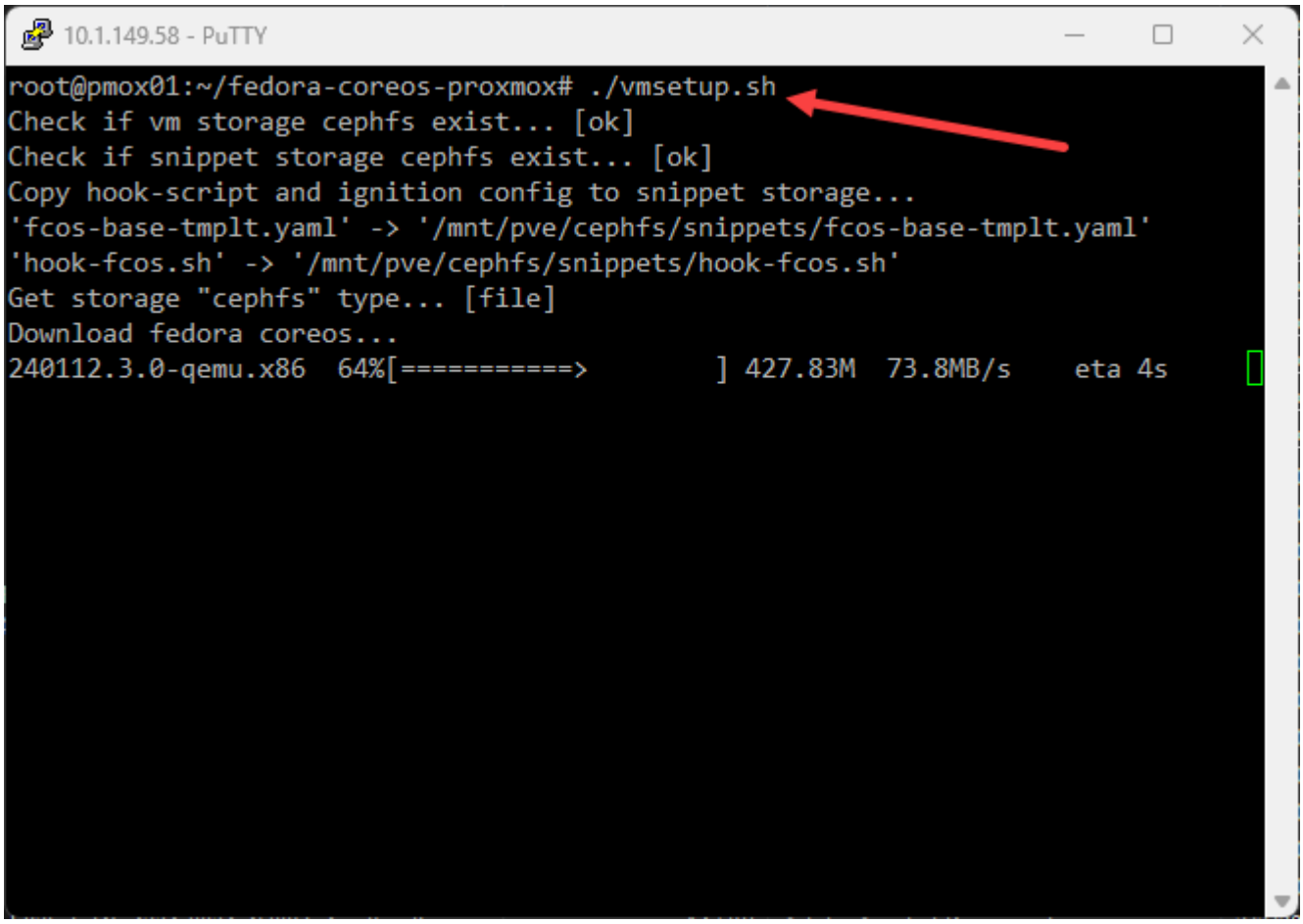


Enabling snippets on the local storage repository

3. Run the included shell script

Now that we have the snippets enabled on the storage of our choice, we can run the vmsetup.sh script included in the cloned repo.

It will automatically download the Fedora CoreOS image in the QEMU QCOW format needed. The Fedora CoreOS container templates in Proxmox streamline the deployment process. The Fedora CoreOS template allows for new container creations and uses the configuration files for settings specific to Fedora CoreOS to ensure smooth operation within the Proxmox environment.



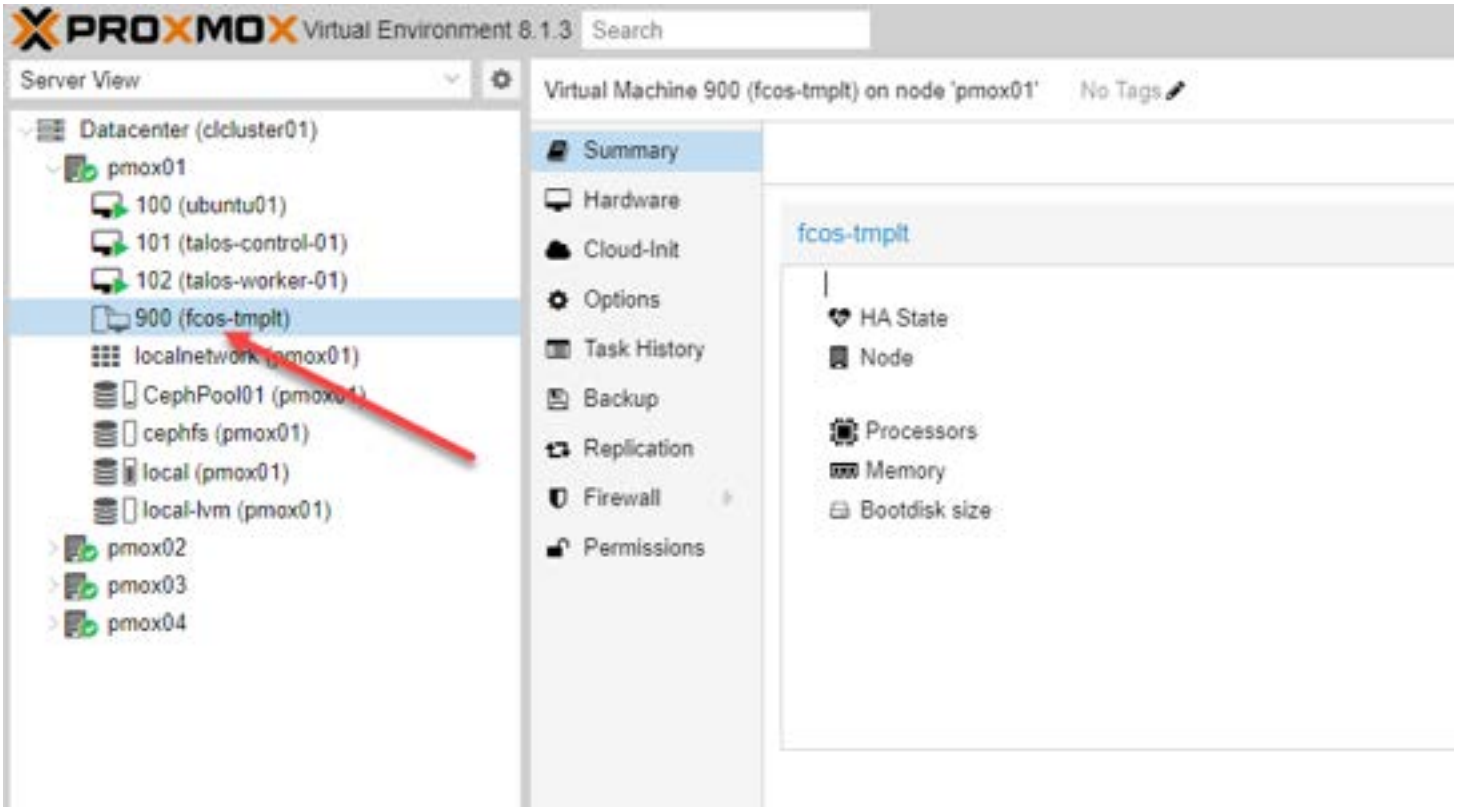
Running the vmsetup.sh script

The process should complete with the message at the bottom: **Convert VM 900 in proxmox vm template**.

```
10.1.149.58 - PuTTY
transferred 8.4 GiB of 10.0 GiB (83.91%)
transferred 8.5 GiB of 10.0 GiB (84.94%)
transferred 8.6 GiB of 10.0 GiB (85.95%)
transferred 8.7 GiB of 10.0 GiB (87.07%)
transferred 8.8 GiB of 10.0 GiB (88.14%)
transferred 8.9 GiB of 10.0 GiB (89.16%)
transferred 9.0 GiB of 10.0 GiB (90.22%)
transferred 9.1 GiB of 10.0 GiB (91.23%)
transferred 9.2 GiB of 10.0 GiB (92.30%)
transferred 9.3 GiB of 10.0 GiB (93.32%)
transferred 9.4 GiB of 10.0 GiB (94.35%)
transferred 9.5 GiB of 10.0 GiB (95.37%)
transferred 9.6 GiB of 10.0 GiB (96.37%)
transferred 9.7 GiB of 10.0 GiB (97.39%)
transferred 9.8 GiB of 10.0 GiB (98.44%)
transferred 9.9 GiB of 10.0 GiB (99.44%)
transferred 10.0 GiB of 10.0 GiB (100.00%)
transferred 10.0 GiB of 10.0 GiB (100.00%)
Successfully imported disk as 'unused0:CephPool01:vm-900-disk-0'
update VM 900: -scsi0 CephPool01:vm-900-disk-0,discard=on -scsihw virtio-scsi-pc
i
update VM 900: -hookscript cephfs:snippets/hook-fcos.sh
Convert VM 900 in proxmox vm template... [done]
root@pmox01:~/fedora-coreos-proxmox#
```

The vm is converted to a template

If you hop over to the Proxmox web interface, you will see the new virtual machine template.

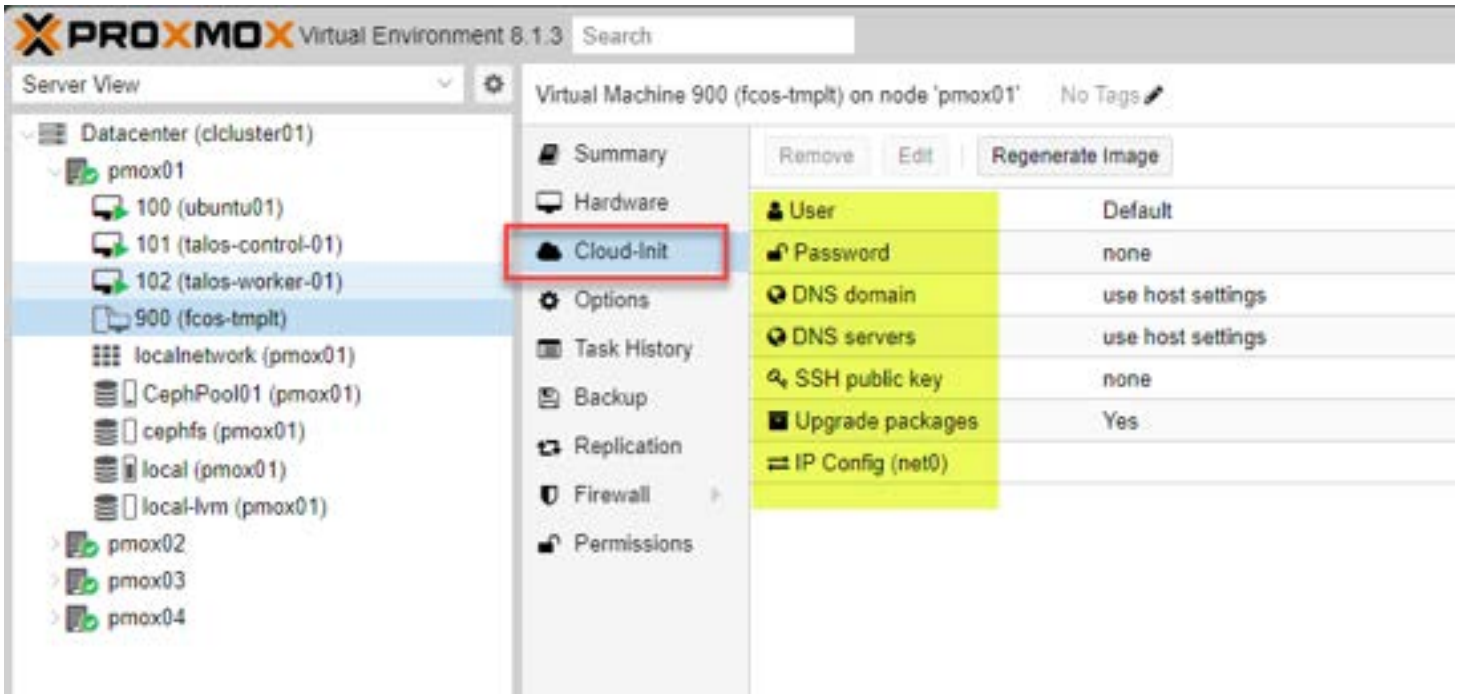


The new template vm

4. Configure the cloud-init settings for the created template

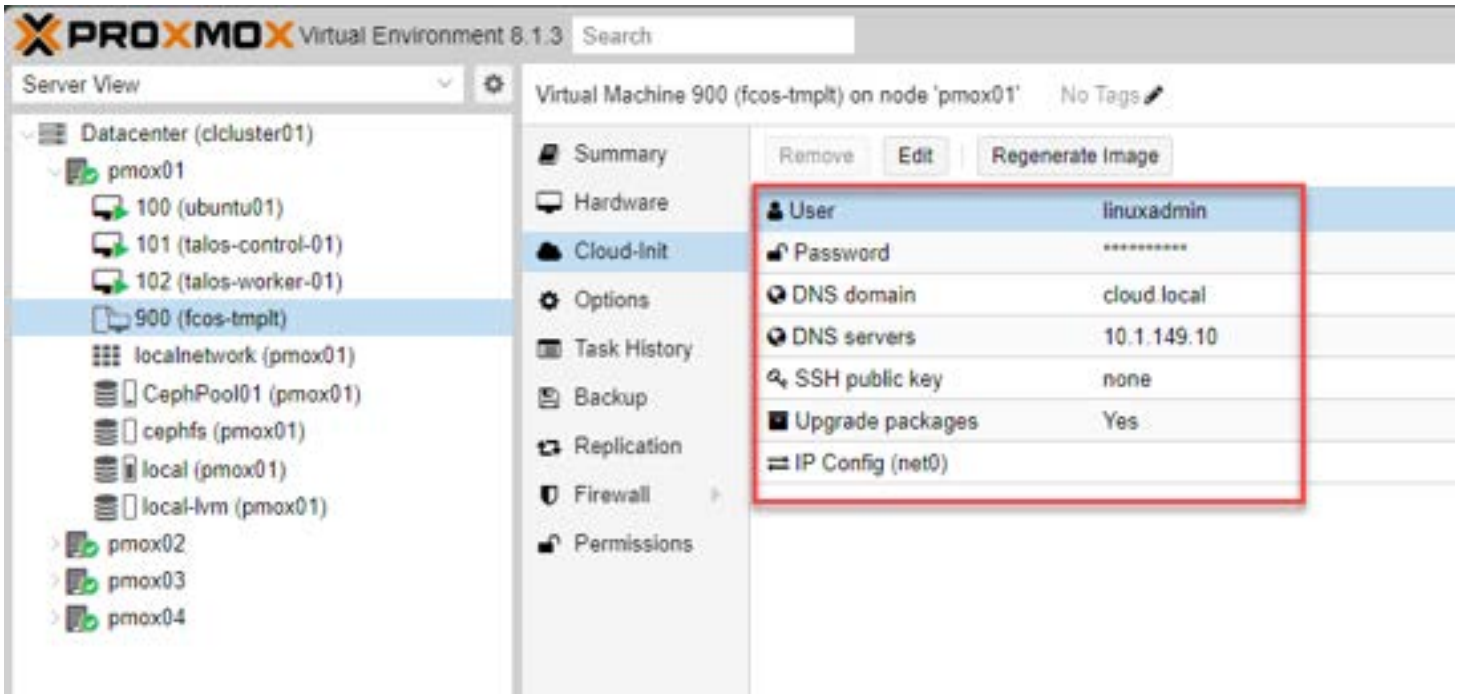
Click the VM and then look at the **Cloud-Init** settings. Here you will find settings you can customize for:

- User
- Password (passwd)
- DNS domain
- DNS servers
- SSH public key
- Upgrade packages
- IP Config (defaults to the default Linux bridge)



Configuring cloud init parameters for proxmox fedora coreos install

Below, I have configured custom settings for Cloud-Init.

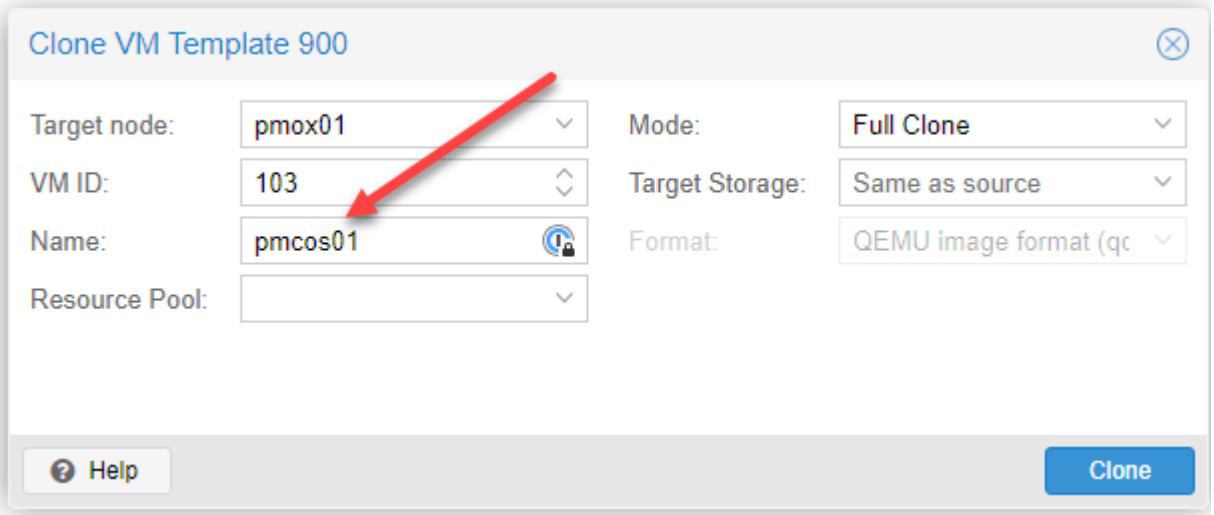


Entering custom cloud init parameters for fedora coreos in proxmox

5. Clone the template VM to a new Fedora CoreOS virtual machine

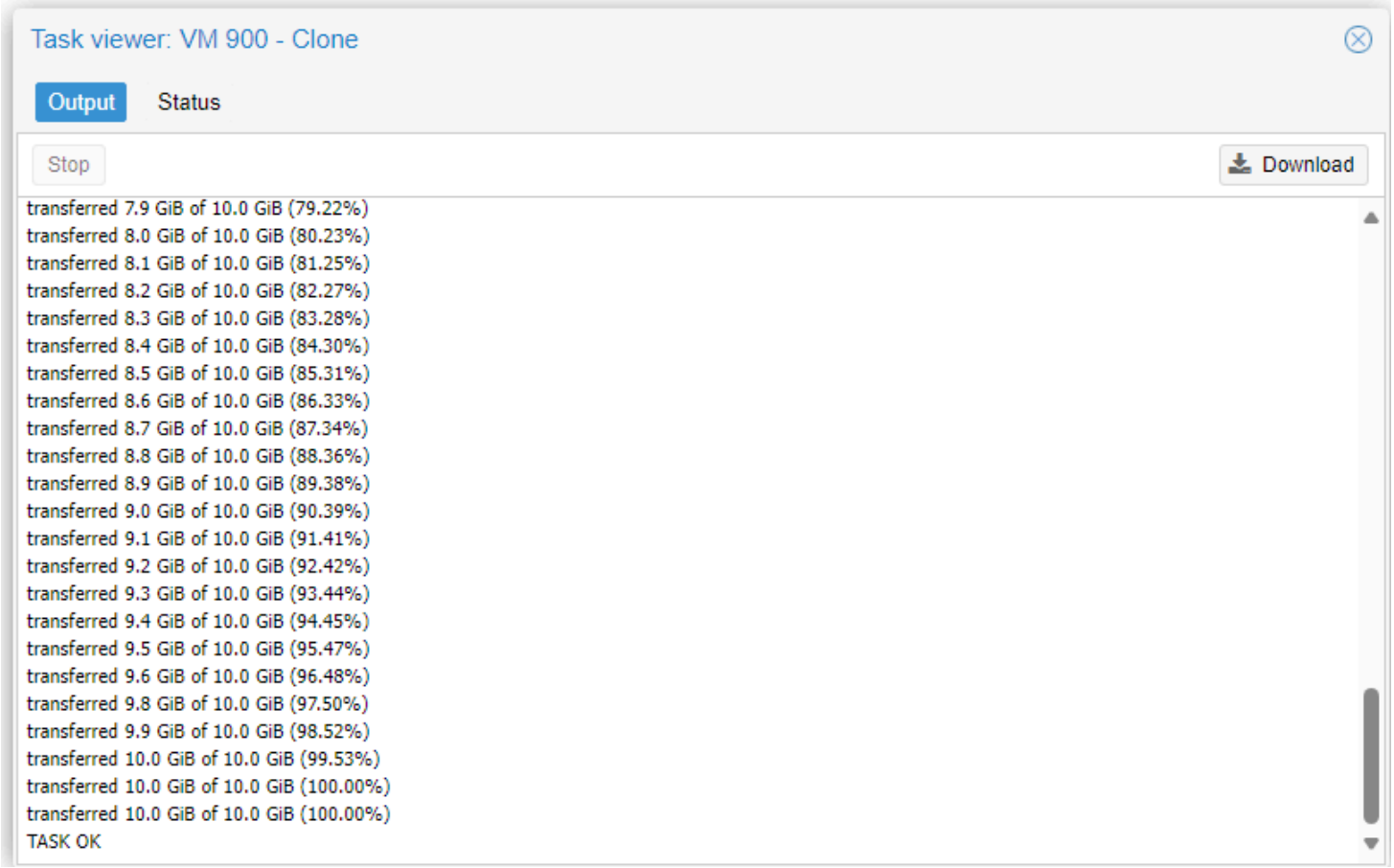
Now that we have the Cloud-Init settings configured, we can clone a new virtual machine from the new template.

Cloning a new VM called **pmcos01** from the newly created template.



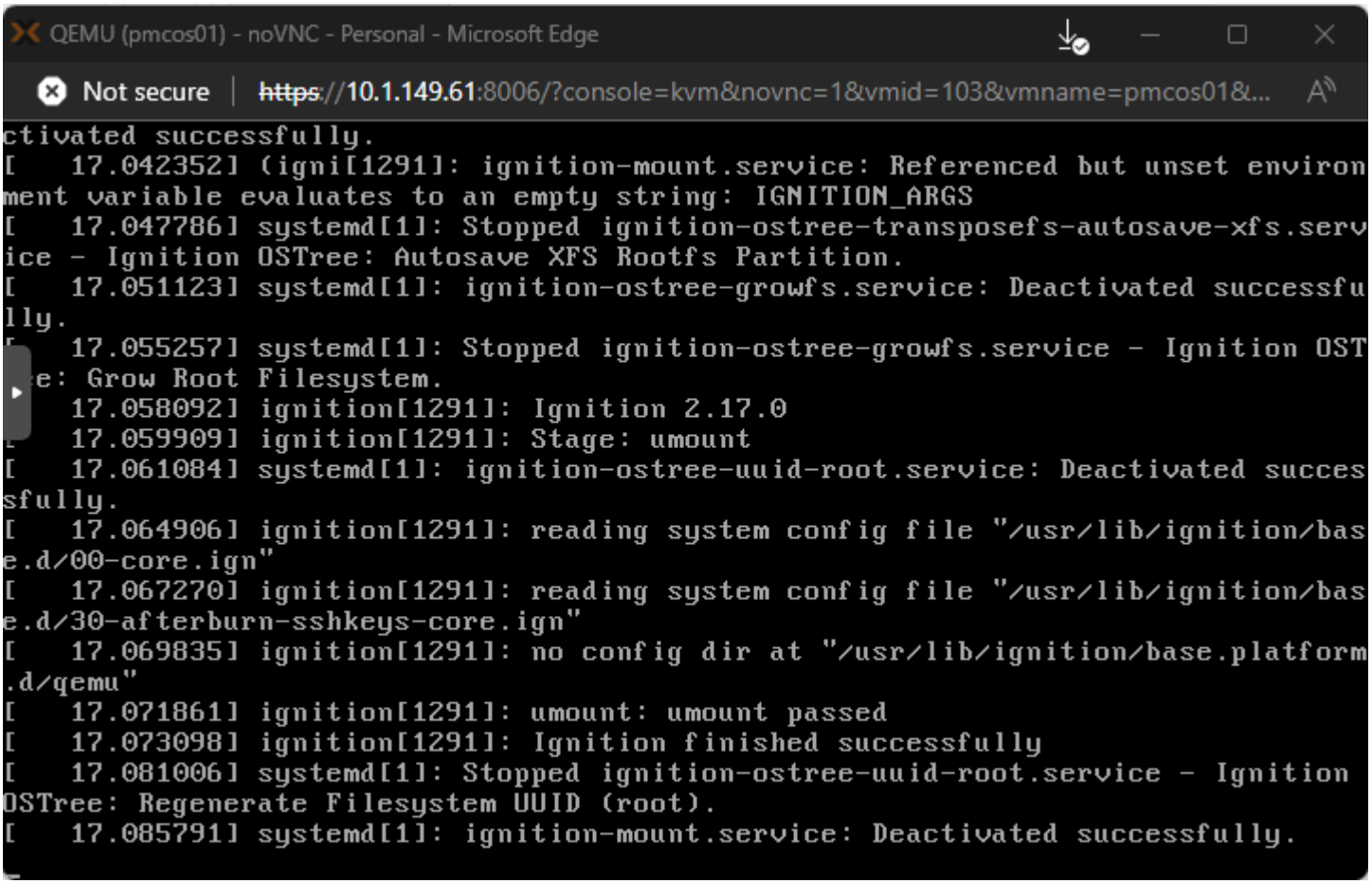
Cloning a new virtual machine for coreos installation

The clone task completes successfully. As you can see, the process to spin up quick Dev workloads for app development, websites, working with source, etc, is easy.



The cloning task is successful for cloning a new proxmox fedora coreos installation

Now, we boot the new virtual machine and it boots. We can see the OS loading the config from the Ignition file.



Booting the new fedora coreos installation in proxmox

After the machine fully boots and grabs an IP address, I log into the VM on the console, and I used the **linuxadmin** user I had specified in the cloud-init settings.

Success! I can login with the new linuxadmin user, showing the VM has used our cloud-init settings.

```
Fedora CoreOS 39.20240112.3.0
Kernel 6.6.9-200.fc39.x86_64 on an x86_64 (tty1)

SSH host key: SHA256:p12X223leTWN9E4dLz9vgBKu/XU6Ka0DlsOSLVwJXkg (ED25519)
SSH host key: SHA256:xe8PhGPDfVoi6fEK2QzrVxUWkAmUreHK0numBwPx7Bg (ECDSA)
SSH host key: SHA256:wX+xJnCfc6AYIEBwsefQModWIxEYuZSl9lcB+Jfta6c (RSA)
ens18: 10.1.149.203 fe80::7207:2000:f181:6325
Ignition: ran on 2024/02/12 20:38:27 UTC (this boot)
Ignition: user-provided config was applied
No SSH authorized keys provided by Ignition or Afterburn
pmcos01 login: linuxadmin
Password:
Fedora CoreOS 39.20240112.3.0
[systemd]
Failed Units: 1
  geco-motd.service
[linuxadmin@pmcos01 ~]$ _
```

Logging into the fedora coreos virtual machine

The Fedora CoreOS installation already has Docker preinstalled, so we can create containers, including system containers and application containers immediately after cloning over new VMs. Now all we need to do is start spinning up our containers.

```
  swarm       Manage Swarm

Commands:
  attach      Attach local standard input, output, and error streams to a running container
  commit      Create a new image from a container's changes
  cp          Copy files/folders between a container and the local filesystem
  create      Create a new container
  diff        Inspect changes to files or directories on a container's filesystem
  events      Get real time events from the server
  export      Export a container's filesystem as a tar archive
  history     Show the history of an image
  import      Import the contents from a tarball to create a filesystem image
  inspect     Return low-level information on Docker objects
  kill        Kill one or more running containers
  load        Load an image from a tar archive or STDIN
  logs        Fetch the logs of a container
  pause       Pause all processes within one or more containers
  port        List port mappings or a specific mapping for the container
  rename      Rename a container
  restart     Restart one or more containers
  rm          Remove one or more containers
  rmi         Remove one or more images
  save        Save one or more images to a tar archive (streamed to STDOUT by default)
  start       Start one or more stopped containers
  stats       Display a live stream of container(s) resource usage statistics
  stop        Stop one or more running containers
  tag         Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  wait        Block until one or more containers stop, then print their exit codes

Global Options:
      --config string      Location of client config files (default "/var/home/core/.docker")
  -c, --context string     Name of the context to use to connect to the daemon (overrides DOCKER_HOST
                           context use")
  -D, --debug              Enable debug mode
  -H, --host list          Daemon socket to connect to
  -l, --log-level string   Set the logging level ("debug", "info", "warn", "error", "fatal") (default
      --tls                Use TLS; implied by --tlsverify
      --tlscacert string   Trust certs signed only by this CA (default "/var/home/core/.docker/ca.pem"
      --tlscert string     Path to TLS certificate file (default "/var/home/core/.docker/cert.pem")
```

Fedora coreos comes out of the box ready to run docker containers

FAQs on Integrating Fedora CoreOS with Proxmox

How does Fedora CoreOS improve container security in Proxmox?

Fedora CoreOS applies SELinux and auto-updates to enhance security. It isolates containers effectively, using the host kernel safely, ensuring a secure container environment within Proxmox.

Why choose Fedora CoreOS for Proxmox containers?

Fedora CoreOS's minimal design and auto-update capabilities make it ideal for Proxmox, ensuring a lightweight, secure base for containers. Its compatibility with container orchestration tools like Kubernetes simplifies management.

Can Docker containers be migrated to Fedora CoreOS on Proxmox?

Yes. Fedora CoreOS supports Docker, allowing for a smooth transition of Docker containers to your Proxmox setup, maintaining flexibility across different container technologies.

What role do container templates play in deploying Fedora CoreOS on Proxmox?

Proxmox's container templates provide ready-to-use Fedora CoreOS images, simplifying setup. They enable quick deployment, ensuring containers are configured with the necessary settings from the start.

What are LXC containers in Proxmox?

LXC containers are a Linux container instance that provide a very “full operating system-like” experience” without the need to run a full virtual machine for running other operating systems.

Managing Fedora CoreOS container storage and network in Proxmox?

Proxmox allows for easy storage and network adjustments via its web interface or CLI. For Fedora CoreOS containers, settings can be tailored during setup or altered later to meet changing demands.

Ensuring smooth Fedora CoreOS updates in Proxmox?

Keep Fedora CoreOS templates updated and watch for new releases. Automatic updates in Fedora CoreOS help keep your system secure with minimal manual effort.

Is Fedora CoreOS as efficient as other Linux distros on Proxmox?

Yes, Fedora CoreOS is designed for containers, making it equally or more efficient than traditional Linux distributions in Proxmox environments by optimizing resource use.

Fedora CoreOS backup and recovery strategies in Proxmox?

Leverage Proxmox’s backup tools or integrate third-party solutions to secure Fedora CoreOS containers, ensuring data protection and quick recovery in case of data loss.

Limitations of using Fedora CoreOS for application containers?

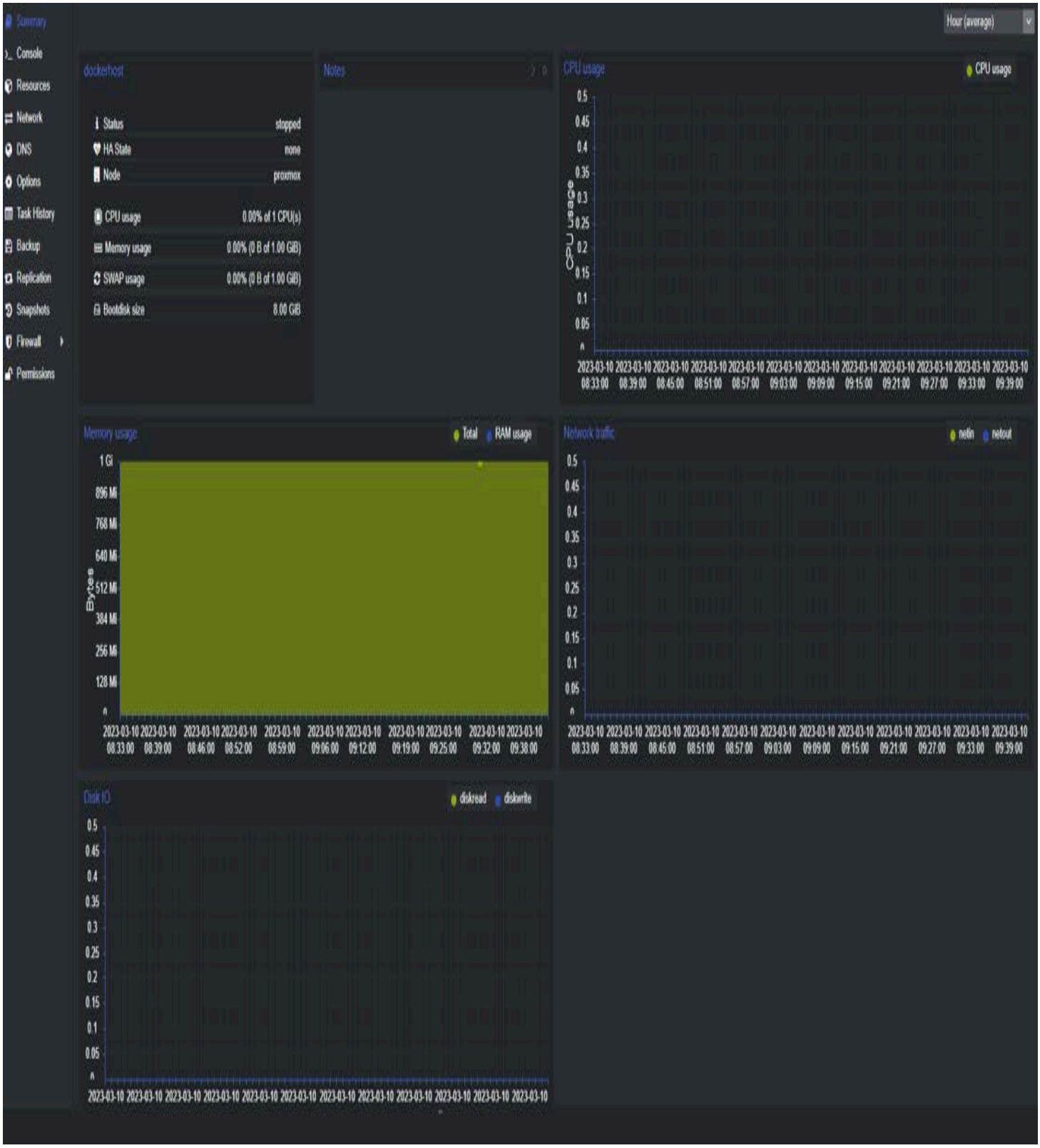
Specific application needs might highlight Fedora CoreOS limitations and require troubleshooting, such as software compatibility or resource requirements. Evaluating these aspects early helps tailor the Proxmox environment to your needs.

Wrapping up Proxmox containers Fedora CoreOS install

There are many advantages of using Fedora CoreOS with Proxmox for managing containers and virtual machines. Proxmox is gaining popularity in the home lab and even the business realm. Especially with the recent Broadcom shakeup with the VMware product portfolio, I think many others will be switching over to Proxmox and other options if they are currently running VMware. Fedora CoreOS provides the resources needed to go all in on an operating system set for container mode, giving dev and DevOps users what they need for development and a platform for running containers across the board. You don’t have to have a subscription for either and you can freely use many of the forum, wiki, and support thread resources out there. Let me know in the comments what you think about Fedora CoreOS and also be sure to sign up for the forums.

Proxmox Helper Scripts you can use

March 10, 2023
Proxmox



4d8096b1 b59f 49b8 9c45 b9bdc364450c

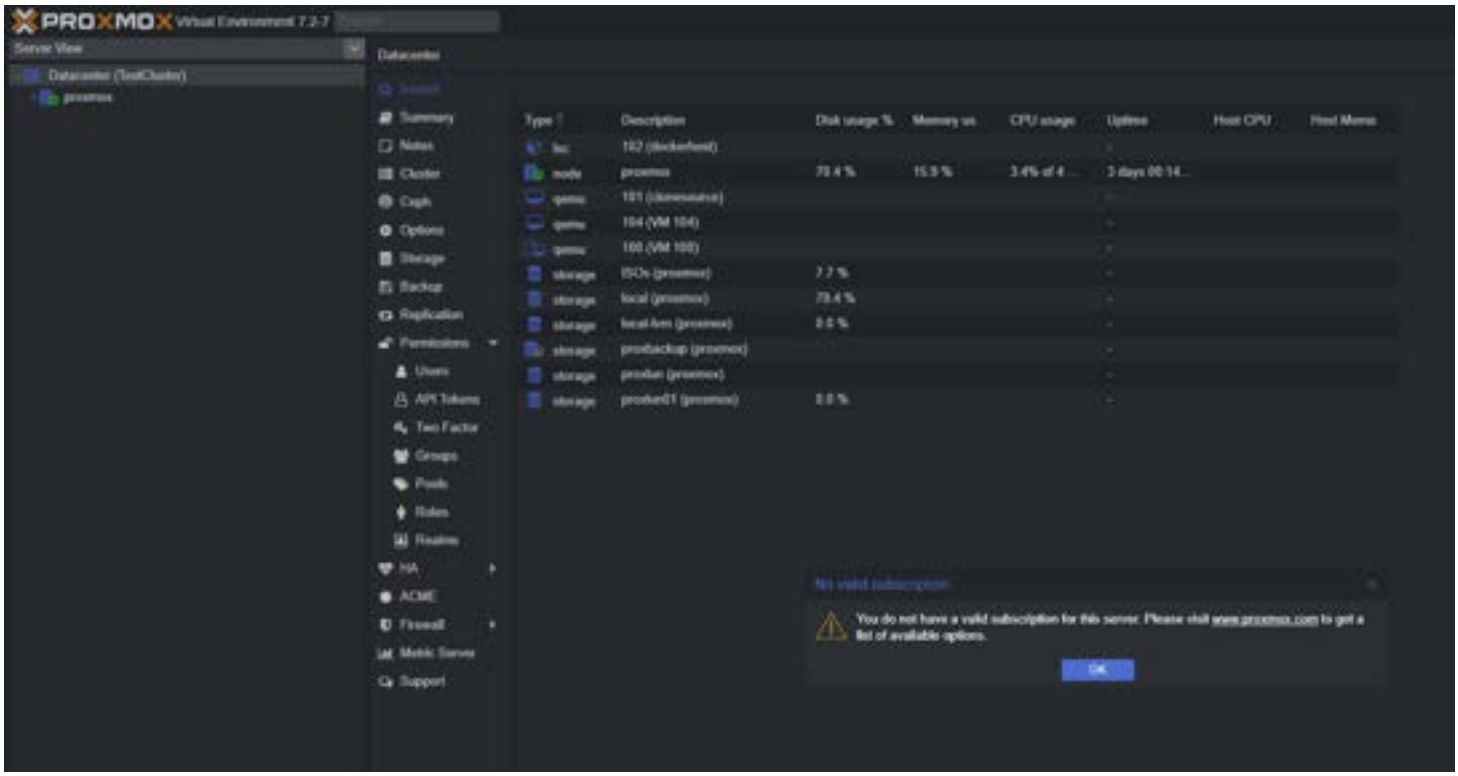
Proxmox is an open-source virtualization platform that allows users to create and manage virtual machines and containers. One of the benefits of Proxmox is its ability to automate tasks using helper scripts. Helper scripts are small programs that automate routine tasks, such as backups and migrations, and make managing it much easier.

This blog post will discuss Proxmox scripts, how they work, and some examples of Proxmox helper scripts you can use to automate tasks in your environment.

Table of contents

What are Scripts?

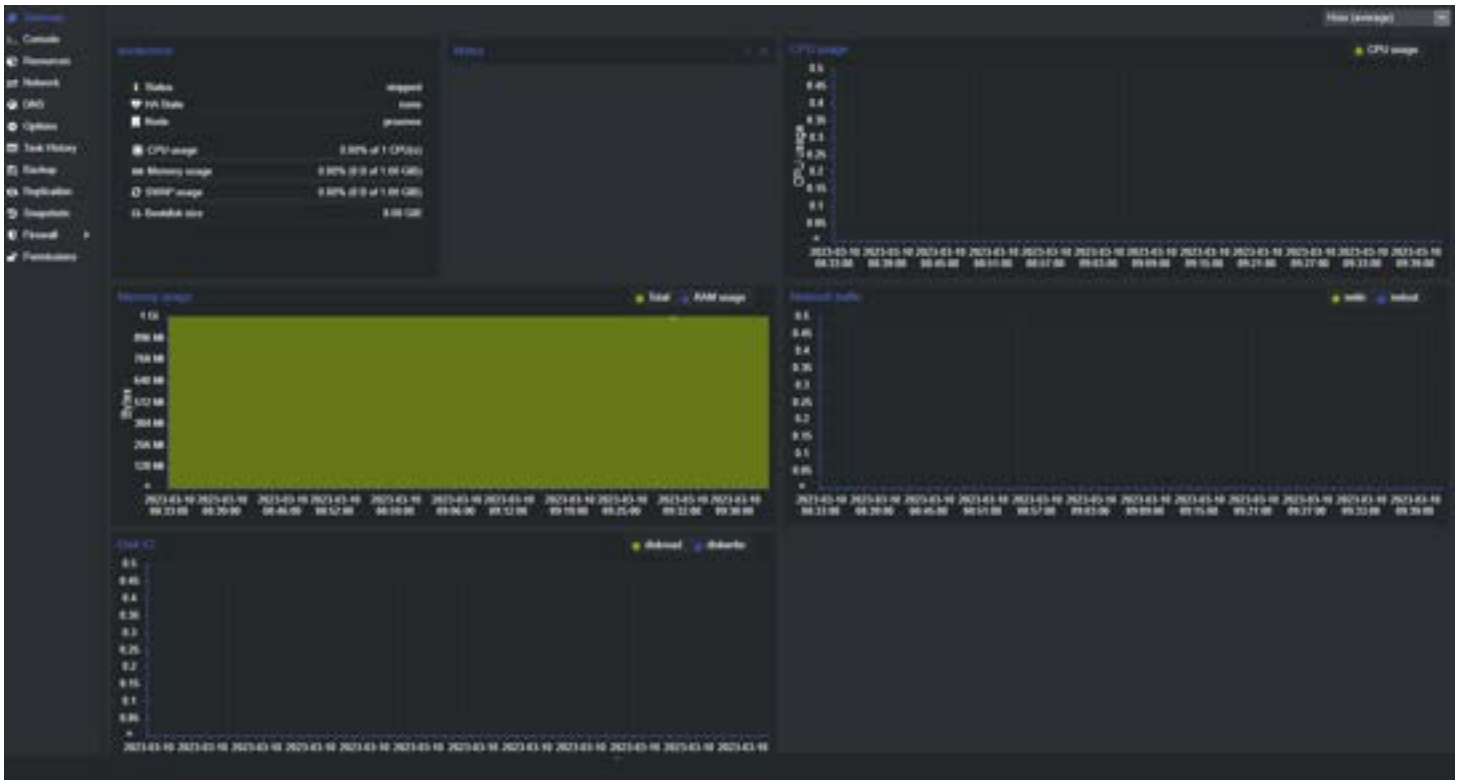
Scripts are small programs that automate tasks in an environment. These scripts can be written in various programming languages, including Bash, Python, and Perl. Helper scripts can perform various automation tasks, such as creating backups, migrating virtual machines, and managing network configurations.



Helper scripts are typically run from the command line and can be run manually or scheduled to run automatically at specific times. Helper scripts can also be integrated with other tools like monitoring software to provide additional functionality.

How do Scripts work?

Proxmox scripts work by interacting with the API. The API is a RESTful API that allows users to interact with the Proxmox environment programmatically. Helper scripts can use the API to perform various tasks, such as creating virtual machines, modifying network configurations, and managing backups.



Helper scripts can be run from the host or another machine on the network and allows admins to do this with due diligence. When a helper script is run, it typically prompts the user for input, such as the virtual machine's name to be created or the backup file name. Once the necessary information is provided, the script interacts with the API to perform the desired task.

Examples of Scripts

There are many great scripts from third-party sites that are easy to find. Sourcing scripts from blogs, videos, and other resources is a great way to find scripts useful for VM or LXC management in your Proxmox environment. However, there are many other types of useful Proxmox scripts, including the following without the need to install any components or have any other prerequisites installed:

Backup script

One of the most common tasks that users perform in a Proxmox environment is creating backups of virtual machines. The Backup Script automates this task by creating backups of selected virtual machines and storing them in a specified location.

The script prompts the user for the name of the virtual machine to be backed up and the location where the backup should be stored. Once the necessary information is provided, the script uses the API to create a backup of the specified virtual machine and store it in the specified location.

Note the following code examples. Please note that these scripts are just examples and may need modification to work in your specific environment without error when loading. Additionally, it is important to always test scripts in a non-production environment before running them in a production environment.

Backup Script:

```
#!/bin/bash

read -p "Enter the name of the virtual machine to be backed up: " vm_name
read -p "Enter the backup file name: " backup_file

# Backup the specified virtual machine to the specified backup file
qm backup $vm_name $backup_file
```

You can then restore from the backup file.

Migration Script

Another common task in an environment is migrating virtual machines from one host to another. The Proxmox Migration Script automates this task by migrating selected virtual machines from one Proxmox host to another.

The script prompts the user for the name of the virtual machine to be migrated and the name of the target host. Once the necessary information is provided, the script uses the API to migrate the specified virtual machine to the specified target host.

Migration Script:

```
#!/bin/bash

read -p "Enter the name of the virtual machine to be migrated: " vm_name
read -p "Enter the name of the target host: " target_host

# Migrate the specified virtual machine to the specified target host
qm migrate $vm_name $target_host
```

Firewall Script

Proxmox Firewall Script is a script that automates the configuration of the firewall rules efficiently in an environment. The script prompts the user for the IP address and port number to be blocked or allowed. Once the information is provided, the script uses the Proxmox API to configure the firewall rules accordingly.

Firewall Script:

```
#!/bin/bash

read -p "Enter the IP address to be blocked/allowed: " ip_address
read -p "Enter the port number to be blocked/allowed: " port_number
read -p "Enter 'block' to block the IP address/port combination or 'allow' to allow it: " action

if [ $action == "block" ]; then
    # Block the specified IP address and port number
    pvesh set /cluster/firewall/iptables -ipfilter
"in,${ip_address},tcp,dport=${port_number},j=DROP"
    echo "IP address ${ip_address} blocked on port ${port_number}"
elif [ $action == "allow" ]; then
    # Allow the specified IP address and port number
    pvesh set /cluster/firewall/iptables -ipfilter
"in,${ip_address},tcp,dport=${port_number},j=ACCEPT"
    echo "IP address ${ip_address} allowed on port ${port_number}"
else
    echo "Invalid action specified. Please enter 'block' or 'allow'."
fi
```

Benefits of Scripts

Script automation is a powerful tool that automates routine tasks in a Proxmox environment. These scripts can be written in various programming languages, and they interact with the Proxmox API to perform tasks such as creating backups, migrating virtual machines, and managing network configurations.

Users can save time and streamline their workflows by using helper scripts. With the examples in this blog post, you can create helper scripts to automate tasks in your Proxmox environment. You can create custom scripts that meet your specific needs with some programming knowledge.

```
PS C:\Users\Administrator> get-pvenodeshosts

cmdlet Get-PveNodesHosts at command pipeline position 1
Supply values for the following parameters:
Node: proxmox

Response            : @{data=}
StatusCode          : 200
ReasonPhrase        :
IsSuccessStatusCode : True
RequestResource     : /nodes/proxmox/hosts
Parameters          :
Method              : Get
ResponseType        : json
```

Additionally, many online resources provide pre-built scripts that you can use to automate tasks. [Proxmox provides a GitHub repository containing](#) a collection of useful helper scripts you can use as a starting point for your own scripts. Additionally, there are many online communities, such as the forum, where users share their scripts and offer support to others.

Scripts FAQs

Why are helper scripts important? Scripts are a great way to introduce automation into your environment. Using scripting and automated tasks helps to make operations much more streamlined, effective, and repeatable.

What technologies can you use for Scripts? You can use built-in Bash scripting for automation, Ansible configuration management, or even PowerShell works well for automated environments.

```
Administrator: PowerShell

PowerShell 7.3.2

  A new PowerShell stable release is available: v7.3.3
  Upgrade now, or check out the release page at:
    https://aka.ms/PowerShell-Release?tag=v7.3.3

PS C:\Users\Administrator> connect-pvecluster -hostsandports 10.1.149.74:8006 -credentials root@pam -SkipCertificateChec
k

PowerShell credential request
Enter your credentials.
Password for user root@pam: *************


HostName             : 10.1.149.74
Port                 : 8006
SkipCertificateCheck : True
Ticket               : PVE:root@pam:640B52A3::T9ea6HttV+4wMHfwZYlwI5mjW+I44Sa1OtfuZjqOtD8qjMuAmtkIM+psFa5cXMamLx5z4Wr5E
                       1varN2sNJ1xL673JxXv2uj+v/ZvA5c8ygQbdnODo+ishVOjfEia4H9VAZ7+HXGJRLWUC/5yDrJ34tRNC5fy9c+mkPo2IRgY+
                       rA6tIpbrihSux/CmCCQqpV6igvvDwCiMsSi6rvDpNg5IOSSihV7pGXKuHK4EJI7UUxWxfqvf5zrWIXFJ4yp+ULuTksju/RgV
                       edu8cbTUpsFLjI3G4BuqEo9rUUJLm4dL7CUYDYr/8OIJG2ekhhHIW79hiCxUc/ZzQsLp6lXT71vtw==
CSRFPreventionToken  : 640B52A3:XtE3FQF0onGTBpEZdSL8MHwcl8YnLQcBLPAcHMYvDaE
ApiToken             :


PS C:\Users\Administrator>
```

```
Administrator: PowerShell

PS C:\Users\Administrator> get-pvevm | where {$_.Status -eq "Stopped"}

disk      : 0
maxcpu    : 4
netin     : 0
node      : proxmox
uptime    : 0
id        : qemu/100
maxdisk   : 64424509440
status    : stopped
diskwrite : 0
maxmem    : 4294967296
cpu       : 0
mem       : 0
netout    : 0
template  : 1
name      : VM 100
vmid      : 100
diskread  : 0
type      : qemu

netin     : 0
uptime    : 0
node      : proxmox
status    : stopped
diskwrite : 0
id        : qemu/101
maxdisk   : 34359738368
cpu       : 0
```

Why use Proxmox in your environment? It is a great hypervisor with many features and capabilities for running home labs or production workloads.

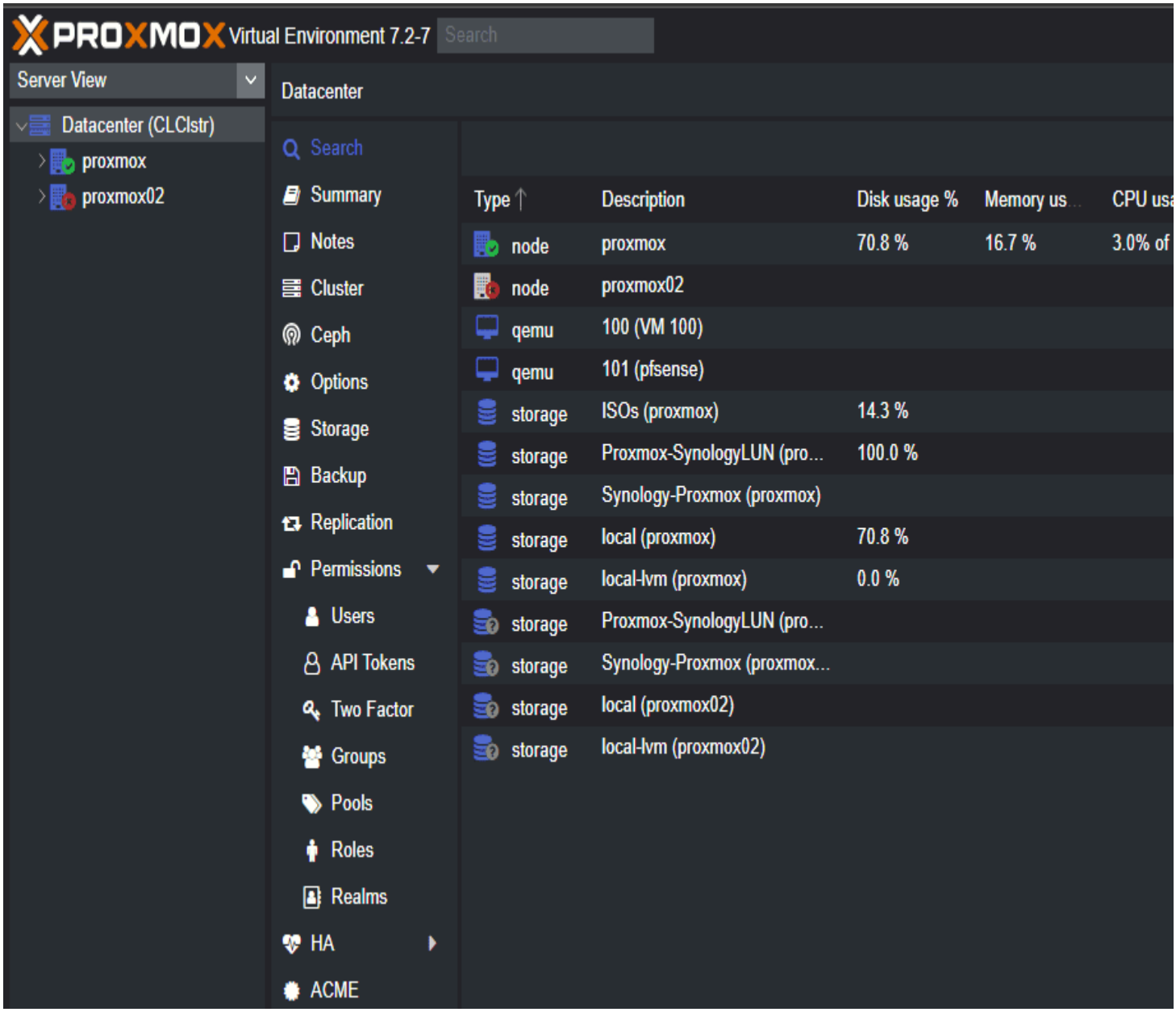
Wrapping up

Proxmox helper scripts are essential for managing and automating tasks in an environment. By leveraging the power of the API, users can create custom scripts that automate routine tasks and save time.

Whether you are a seasoned user or a newcomer to virtualization, learning how to use helper scripts can help you streamline your workflow and get the most out of your Proxmox environment.

Proxmox scripts PowerShell Ansible and Terraform

January 20, 2023
Proxmox

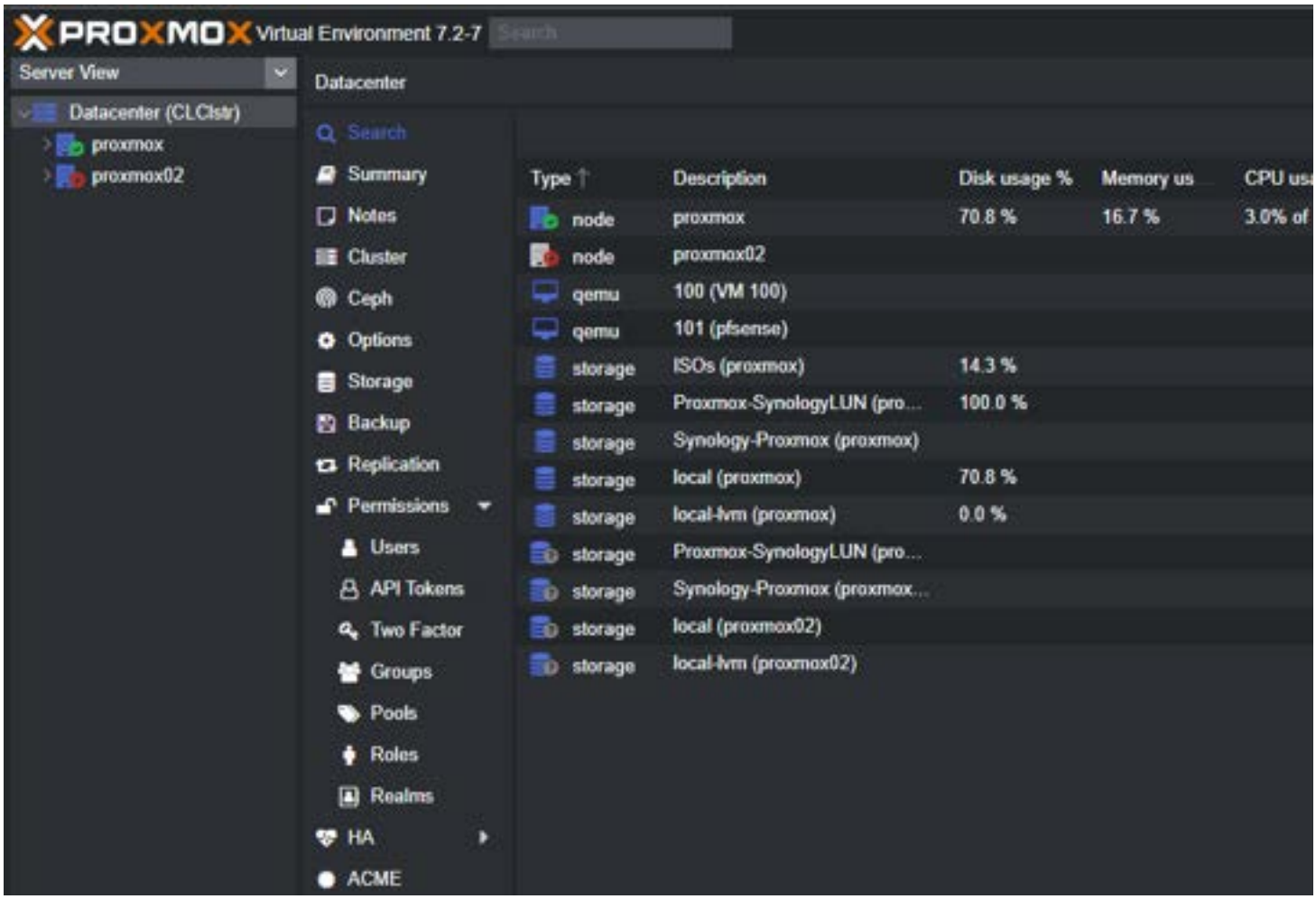


63aff498 be00 458e 9099 5eba0926905c

Proxmox is growing more and more popular, especially for home lab enthusiasts and those looking to spin up labs based on totally free and open-source software. Proxmox has a great API that allows throwing automation tasks at the solution and creating Proxmox helper scripts for automating your Proxmox environment.

Why scripting and automation are important

For many reasons, scripting and automation are essential in today's infrastructure environments. IT admins and DevOps engineers must move quickly and provision, configure, and interact with infrastructure effectively and efficiently. This certainly involves automation.



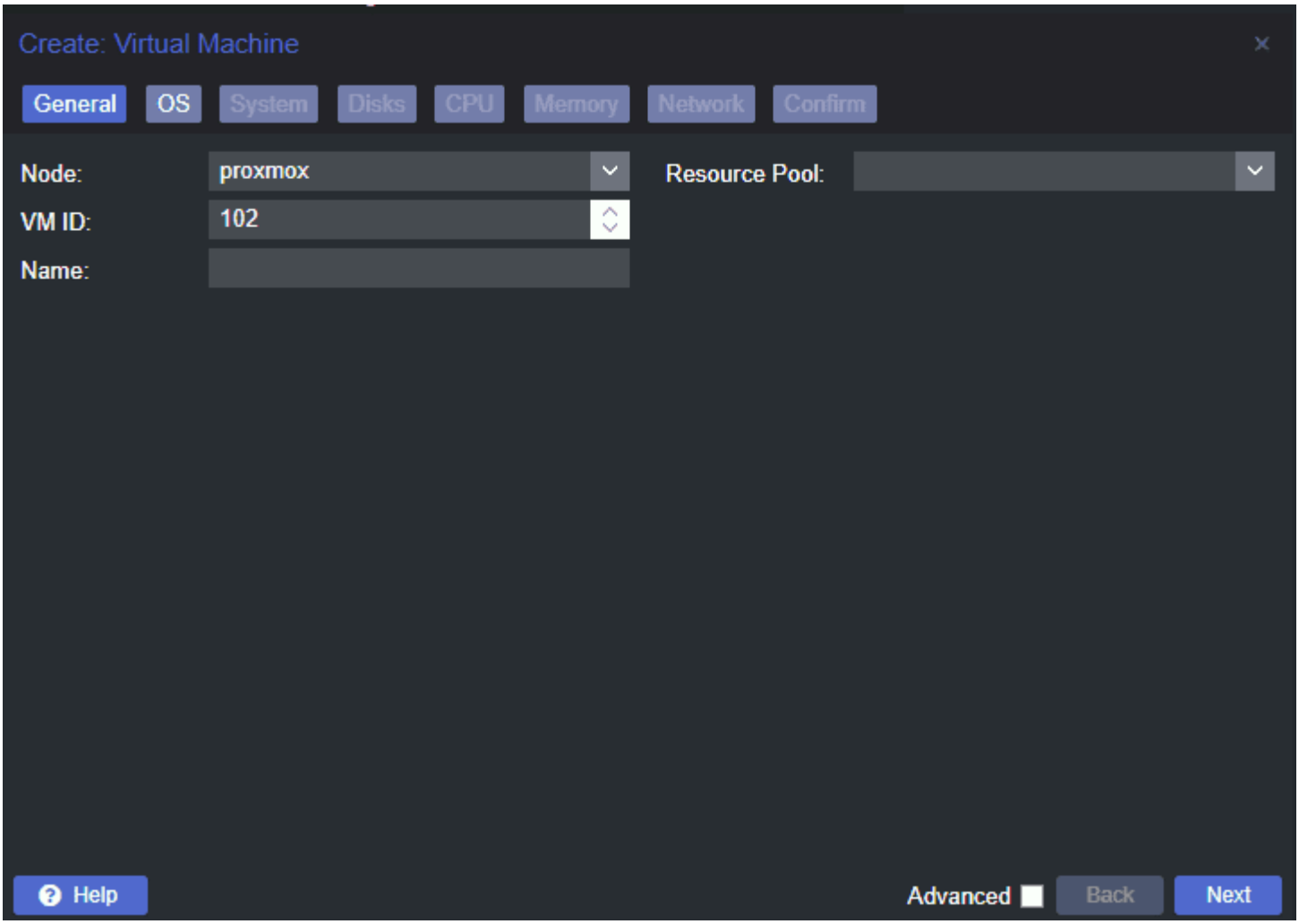
Scripting and automation improve your effectiveness as an administrator and virtualization engineer. If you accept the challenge of learning scripting, it will pay off dividends.

Infrastructure as code

Due to the massive shift to cloud-based technologies, today's infrastructure services is driven by infrastructure as code. It allows admins to commit code to a code repository location, version of that code, and other resources it manages.

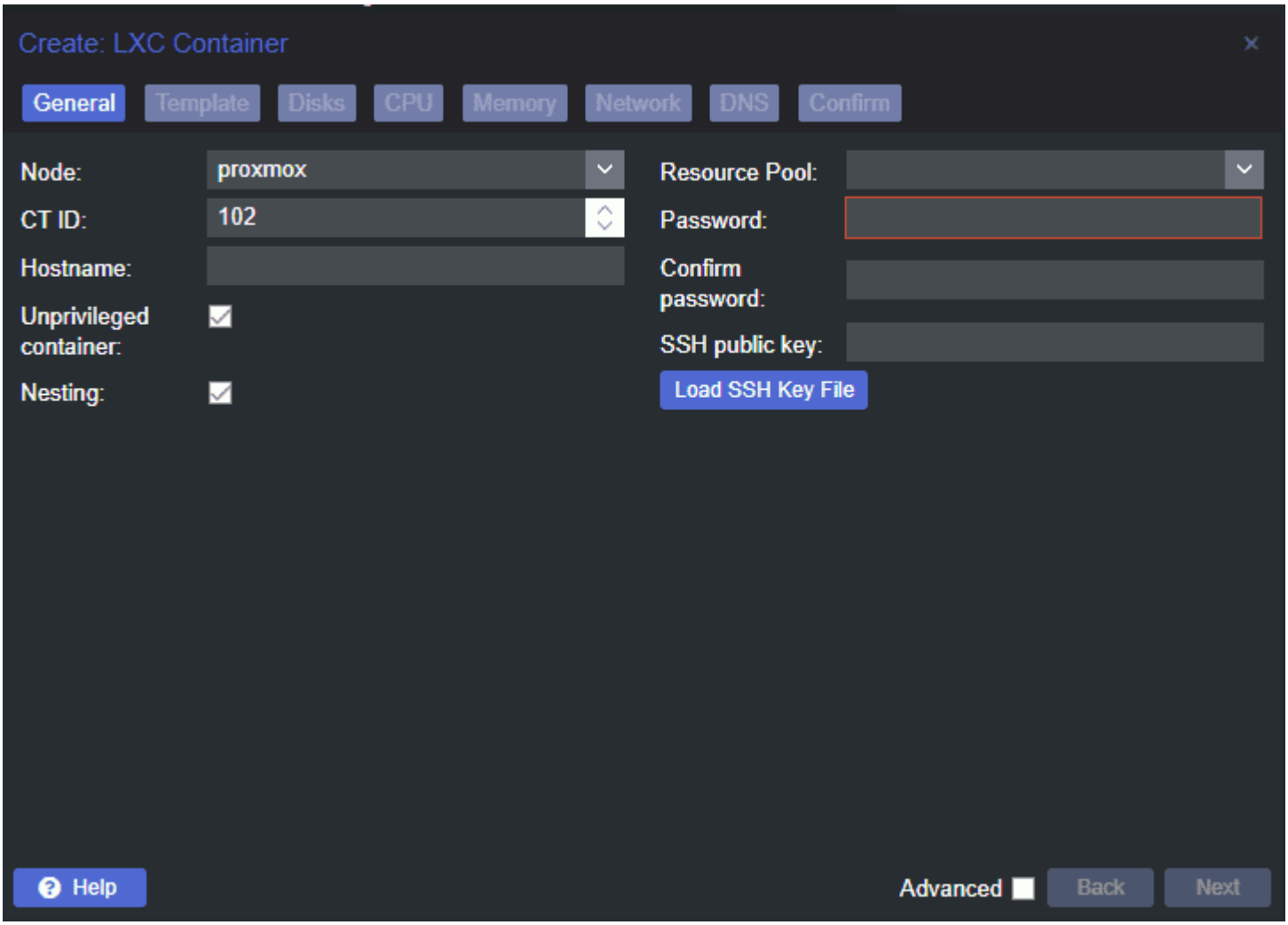
Scripting automation tasks

This infrastructure-as-code approach includes creating VM environments and Docker containers as code. LXC containers can also easily be provisioned in Proxmox VE environments. Using simple and easy script-based approaches, admins can, with due diligence, create scripts to manage the environment.



Don’t reinvent the wheel

Also, there have been so many great scripts and code is already written admins don’t have to reinvent the wheel. Sourcing scripts from free and open-source sites is easy to do, along with other learning tools like YouTube, blog posts, and other third-party sites etc. You can tap into many great learning resources and support forums.



Running into error messages along the way is part of the learning process. However, the effort will outweigh the challenges with massive time and effort savings that automation provides.

Proxmox automated REST API interface

The Proxmox VE solution uses an interface known as RESTful API. The API uses the HTTPS protocol and the server listens to port 8006. So the base URL for that API is: https://your.server:8006/api2/json/

Proxmox VE uses a ticket or token-based authentication. All requests to the API need to include a ticket inside a Cookie (header) or send an API token through the Authorization header.

Proxmox PowerShell scripts

I have found many great repositories working with Proxmox scripts to allow changing settings, install updates, backup your configuration, and loading configurations to give you an idea of what is possible. Check out this Proxmox PowerShell repository, which provides a VMware PowerCLI approach to managing Proxmox with Proxmox helper scripts.

PowerShell Gallery | Corsinvest.ProxmoxVE.Api 7.3.0

This is a great way to create scripts with Proxmox and PowerShell.

```
PS C:\Users\zerouser> Connect-PveCluster -hostsandports 10.1.149.74:8006 -skipcertificatecheck

PowerShell credential request
Proxmox VE Username and password, username formated as user@pam, user@pve, user@yourdomain or user (default domain
pam).
User: root@pam
Password for user root@pam: *************


HostName             : 10.1.149.74
Port                 : 8006
SkipCertificateCheck : True
Ticket               : PVE:root@pam:63D72E82::pqw75uh2oEcfXuvo8E/jLdWhCr7cxVOONILom/mD8bMaRhfHpgyi0i/UWjVNR1rijVSZcOYG1
                       [redacted]
CSRFPreventionToken  : 63D72E82:4t7bUqn7BdoSS9DazKSclWwDfk8OFE57W2MbEAGm63Q
ApiToken             :
```

Connecting to Proxmox with the PowerShell module

Below, we are getting Proxmox VMs using the **get-pvevm** cmdlet.

```
Administrator: PowerShell

PS C:\Users\zerouser> get-pvevm

netin     : 0
diskread  : 0
disk      : 0
template  : 0
diskwrite : 0
name      : VM 100
maxcpu    : 4
mem       : 0
node      : proxmox
maxdisk   : 64424509440
netout    : 0
id        : qemu/100
vmid      : 100
uptime    : 0
type      : qemu
cpu       : 0
status    : stopped
maxmem    : 4294967296
```

Getting Proxmox virtual machines using PowerShell

Proxmox Ansible and Terraform scripts

Proxmox is easy to work with using Ansible and Terraform and also allows great scripting capabilities and functionality. Check out these repositories:

community.general.proxmox module – Management of instances in Proxmox VE cluster — Ansible Documentation

Docs overview | Telmate/proxmox | Terraform Registry

Proxmox helper scripts

Proxmox Helper Scripts | Proxmox Scripts For Home Automation (tteck.github.io)

Wrapping up

Hopefully, this quick guide to Proxmox scripts with PowerShell, Ansible, and Terraform shows there are many great ways to automate Proxmox and create infrastructure as code in your Proxmox VE environment. With the RESTful API driven automation provided by Proxmox, you can create quick and easy infrastructure as code.

Proxmox Backup Server: Ultimate Install, Backup, and Restore Guide

December 4, 2023
Proxmox

Proxmox backup server ultimate guide

Backups are essential to running Proxmox VE in the home lab or production to avoid data loss. Proxmox Backup Server is a free solution to back up and recover Proxmox VE VMs and containers.

Table of contents

What is Proxmox Backup Server?

Proxmox Backup Server (PBS) is a free solution based on the Debian operating system from Proxmox for backing up Proxmox VE virtual machines and container instances. It provides secure Proxmox Backup Server storage and data management features. Using PBS helps protect your critical data from accidental deletion, corruption, ransomware, or other unforeseen events.

Since many commercial enterprise backup solution products don't protect Proxmox, it is great to see that Proxmox has a backup server to protect your Proxmox VE virtual machine workloads. Also, you can run it on your own hardware and select your hardware based on your own resource usage. So, the specific CPU, memory, disk, and hardware platform you choose may vary according to your strategy and needs.

Proxmox Backup Server features

Note the following features:

- **Efficient Data Backup and Recovery:** It provides reliable and quick backup and restoration of virtual machines, containers, and physical hosts. PBS also features Incremental backups. With incremental backups, only the changes changes made since the last backup are stored.
- **Incremental Backup Support:** Only backs up data that has changed since the last backup, to reduce backup time and storage requirements.
- **Data Deduplication:** Reduces the storage space required for backups by only storing unique data blocks.
- **Data Backup Encryption:** uses encryption during transfer and at rest.
- **Web user Interface:** Using a web browser, you can manage backups, restore data, and configure settings.
- **Compression Options:** Supports data compression to further reduce the storage space needed for backups.
- **Snapshot Functionality:** Allows for creating snapshots of data, enabling point-in-time recoveries.
- **ZFS Support:** Integrates with ZFS (Zettabyte File System) for efficient storage management and high data integrity.
- **Flexible Storage Options:** Supports various storage backends, including local directories, NFS targets, and SMB/CIFS.
- **Replication:** You can replicate your backup data to remote sites. This helps to create a 3-2-1 disaster recovery model.
- **Role-Based Access Control:** Allows granular control over user access and permissions.
- **Backup Scheduling:** Automates the backup process through customizable scheduling.
- **Email Notification System:** Sends automated email notifications regarding backup jobs and system status.
- **API for Automation:** Provides a REST API for easy integration with other systems and automation of backup tasks.
- **Support for Multiple Clients:** Compatible with various clients, including Proxmox VE, Linux, and others.
- **Backup Retention Policies:** Customizable retention policies to maintain a balance between storage space and backup availability.
- **Bandwidth Throttling:** Manages network load by controlling the bandwidth used for backup operations.
- **Plugin System for Extensibility:** Supports plugins for extending functionality and integrating with other systems.
- **Backup Verification:** Includes features to verify the integrity of backups, ensuring recoverability.
- **Proxmox VE Integration:** Seamlessly integrates with Proxmox Virtual Environment for centralized management of virtualized infrastructure and backups.

Proxmox Backup Server installation step-by-step instructions

Like installing Proxmox VE virtualization server, installing PBS is extremely easy and looks very much like installing Proxmox VE. Let's install PBS and configure backups of Proxmox virtual machines.

First, you will need to download the PBS release ISO image from Proxmox here: Proxmox Backup Server.

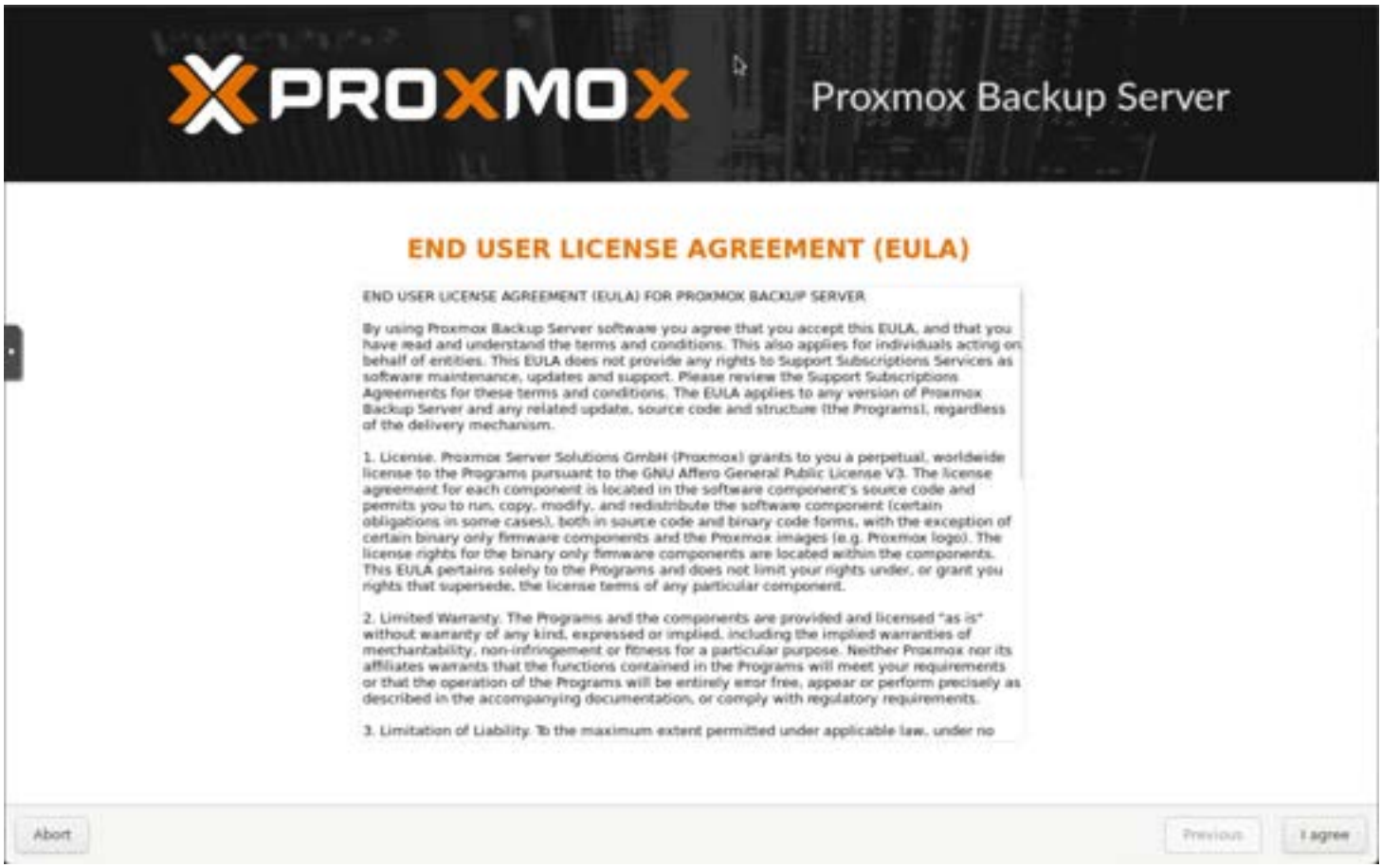
Once you have the ISO file, "burn" the software to a USB flash drive or upload it to your Proxmox VE host if you are hosting your backup server as a virtual machine.

Below is a screenshot of the Proxmox Backup Server virtual machines booting from the ISO installation.

```
Welcome to the Proxmox Backup Server 3.0 installer
initial setup startup
mounting proc filesystem
mounting sys filesystem
boot comandline: BOOT_IMAGE=/boot/linux26 ro ramdisk_size=16777216 rw quiet spla
loading drivers:  i2c_piix4 pata_acpi vmgenid floppy qemu_fw_cfg mac_hid uhci_hc
pkr aesni_intel sha512_ssse3
searching for block device containing the ISO proxmox-backup-server-3.0-1
with ISO ID '5d145420-1501-11ee-a40c-7fb0fe0bd0d3'
testing device '/dev/sr0' for ISO
found Proxmox Backup Server ISO
switching root from initrd to actual installation system
Starting Proxmox installation
Installing additional hardware drivers
Starting hotplug events dispatcher: systemd-udevd.
Synthesizing the initial hotplug events (subsystems)...done.
Synthesizing the initial hotplug events (devices)...done.
Waiting for /dev to be fully populated...done.
mount: devpts mounted on /dev/pts.
 in/dbus-daemon
 arting D-Bus daemon
tempting to get DHCP leases... Internet Systems Consortium DHCP Client 4.4.3-P1
Copyright 2004-2022 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/ens18/bc:24:11:5c:5a:cb
Sending on   LPF/ens18/bc:24:11:5c:5a:cb
Sending on   Socket/fallback
DHCPDISCOVER on ens18 to 255.255.255.255 port 67 interval 6
DHCPOFFER of 10.3.33.230 from 10.3.33.1
DHCPREQUEST for 10.3.33.230 on ens18 to 255.255.255.255 port 67
DHCPACK of 10.3.33.230 from 10.3.33.1
bound to 10.3.33.230 -- renewal in 297690 seconds.
done
Starting chrony for opportunistic time-sync...
Starting a root shell on tty3.
trying to detect country...
```

Running the PBS installer

Accept the end user license agreement (EULA).

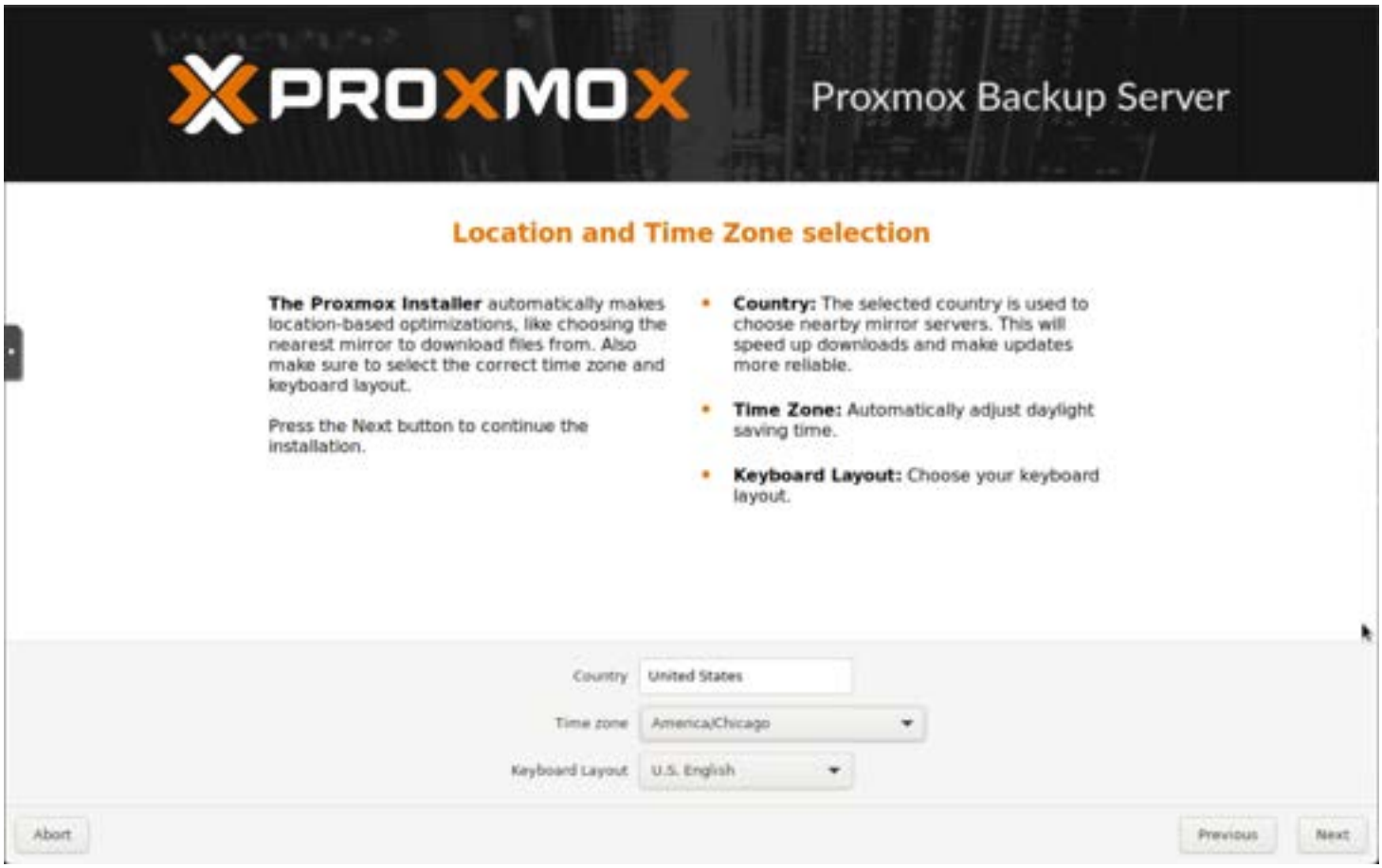


PBS eula

Select the installation drive.

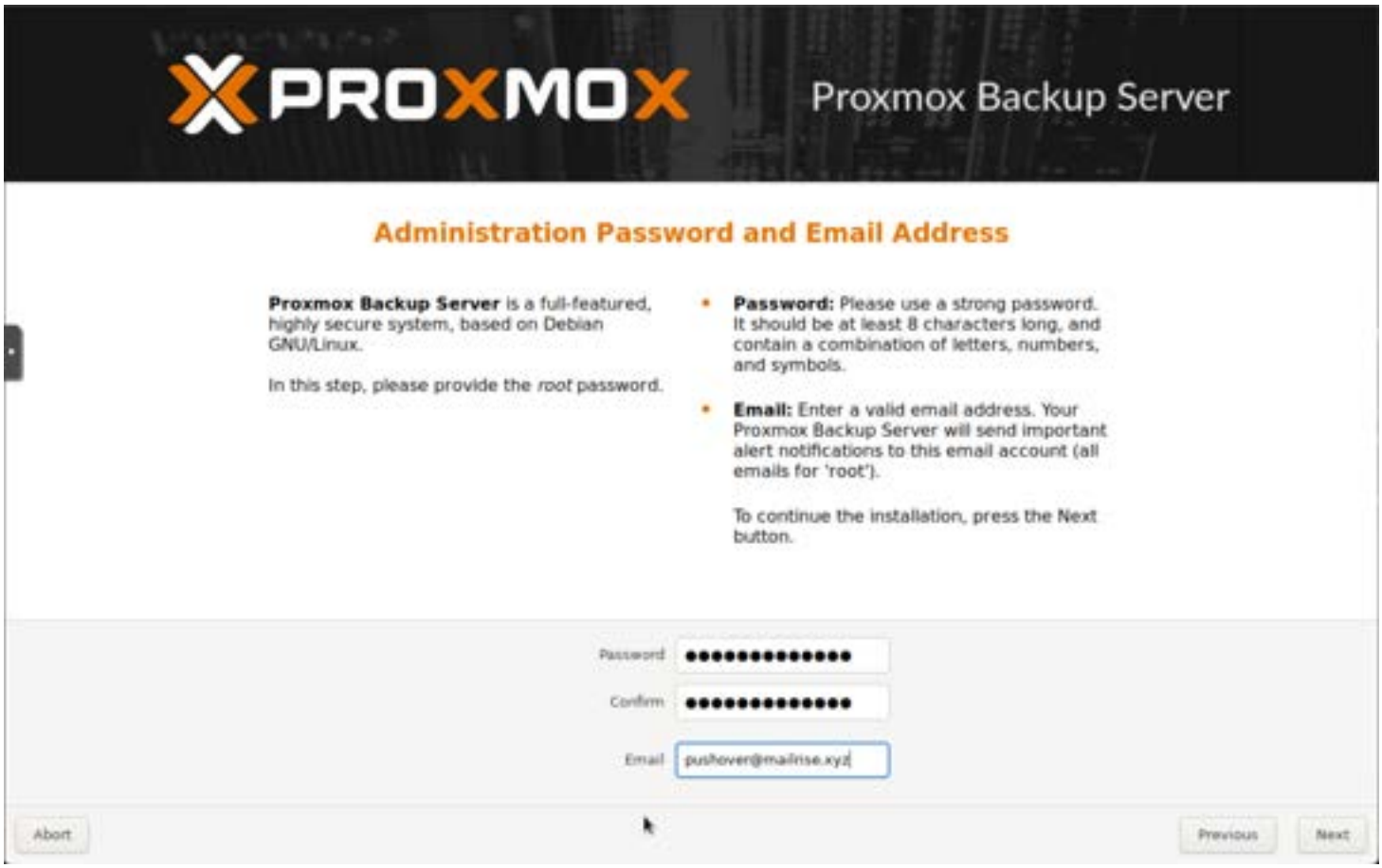
PBS disk configuration

Set your country, time zone, and keyboard layout language.



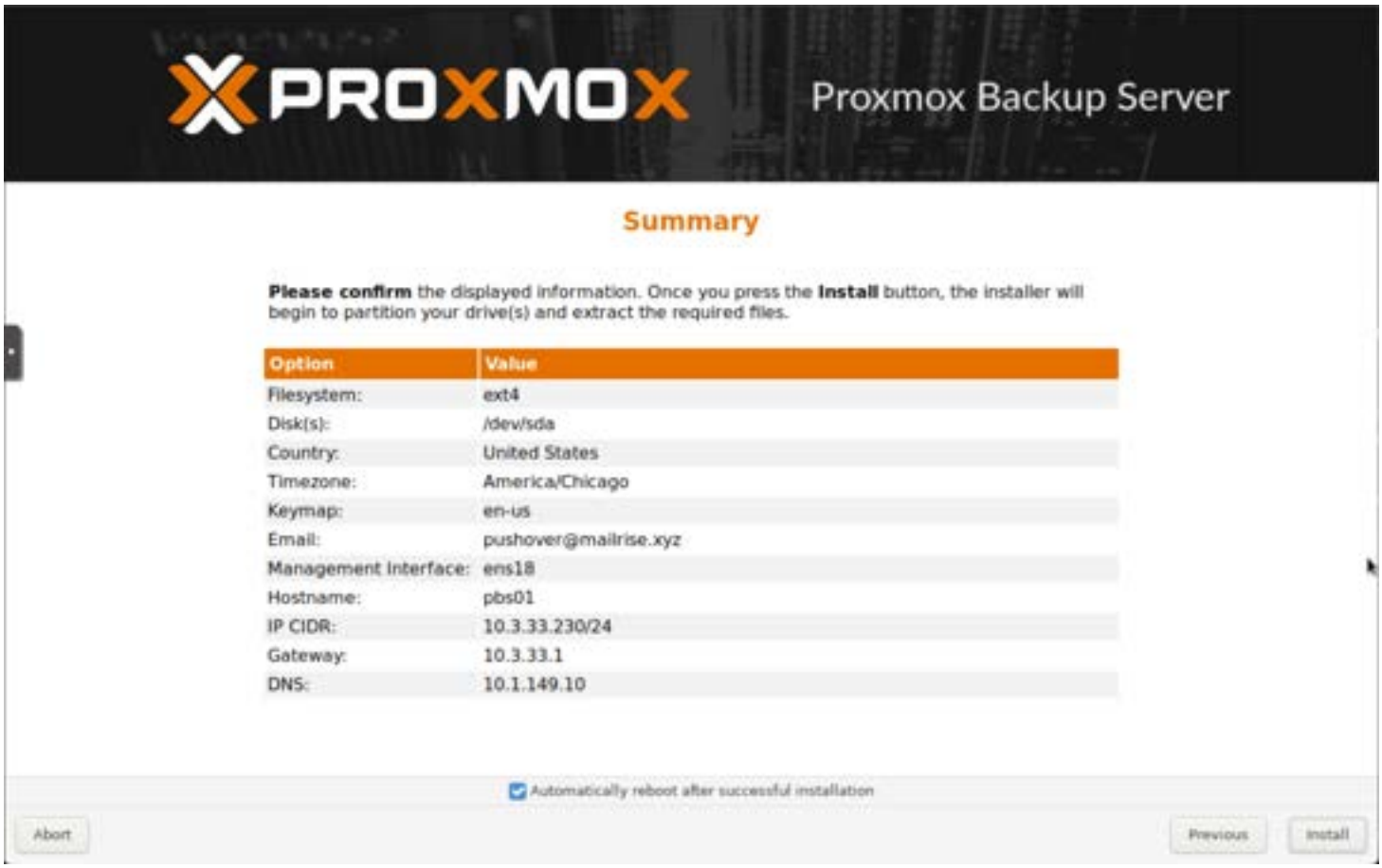
Location timezone and keyboard layout

Create an administrative password and email address.



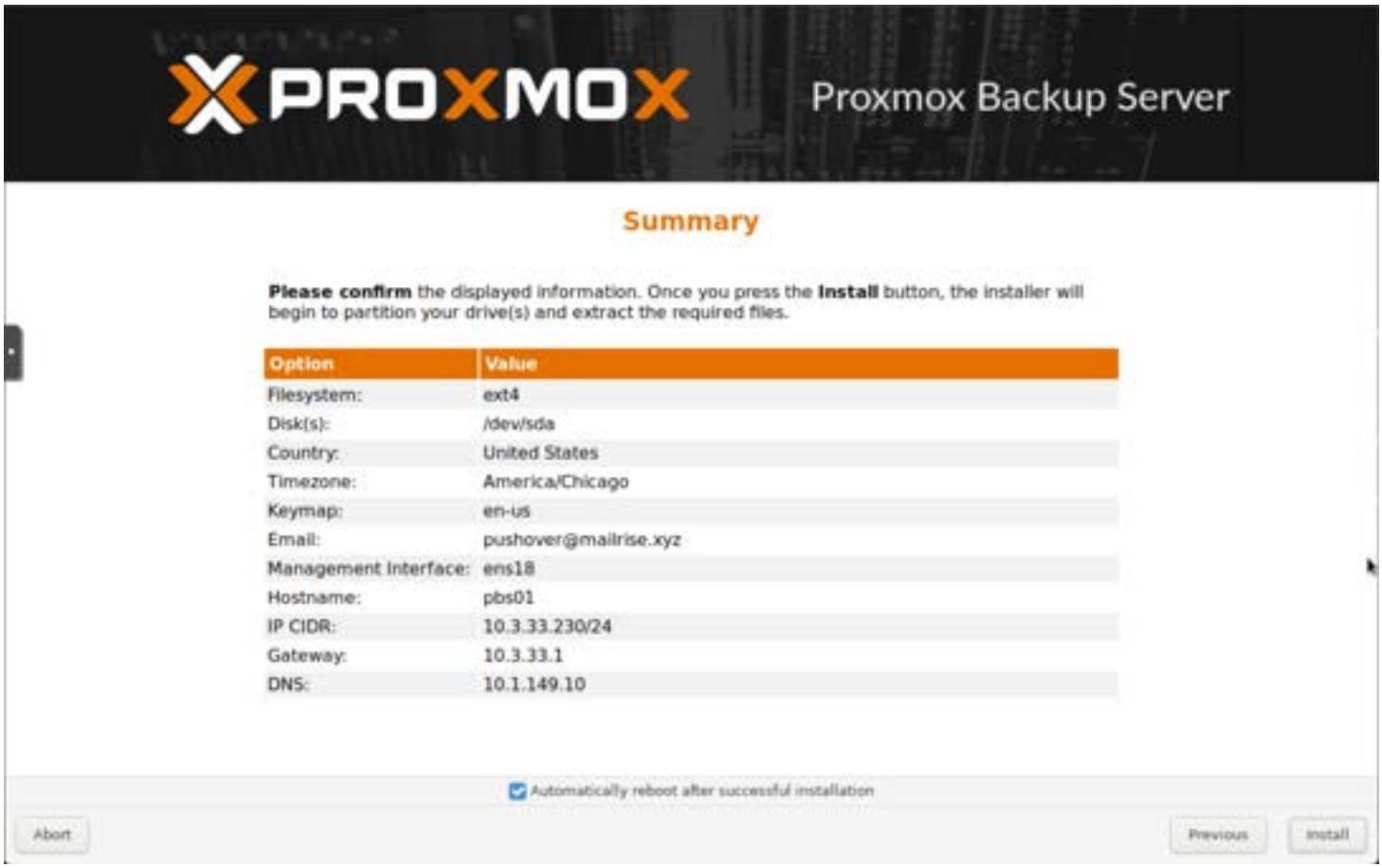
Setting the password and email address

Enter the FQDN and IP address configuration.



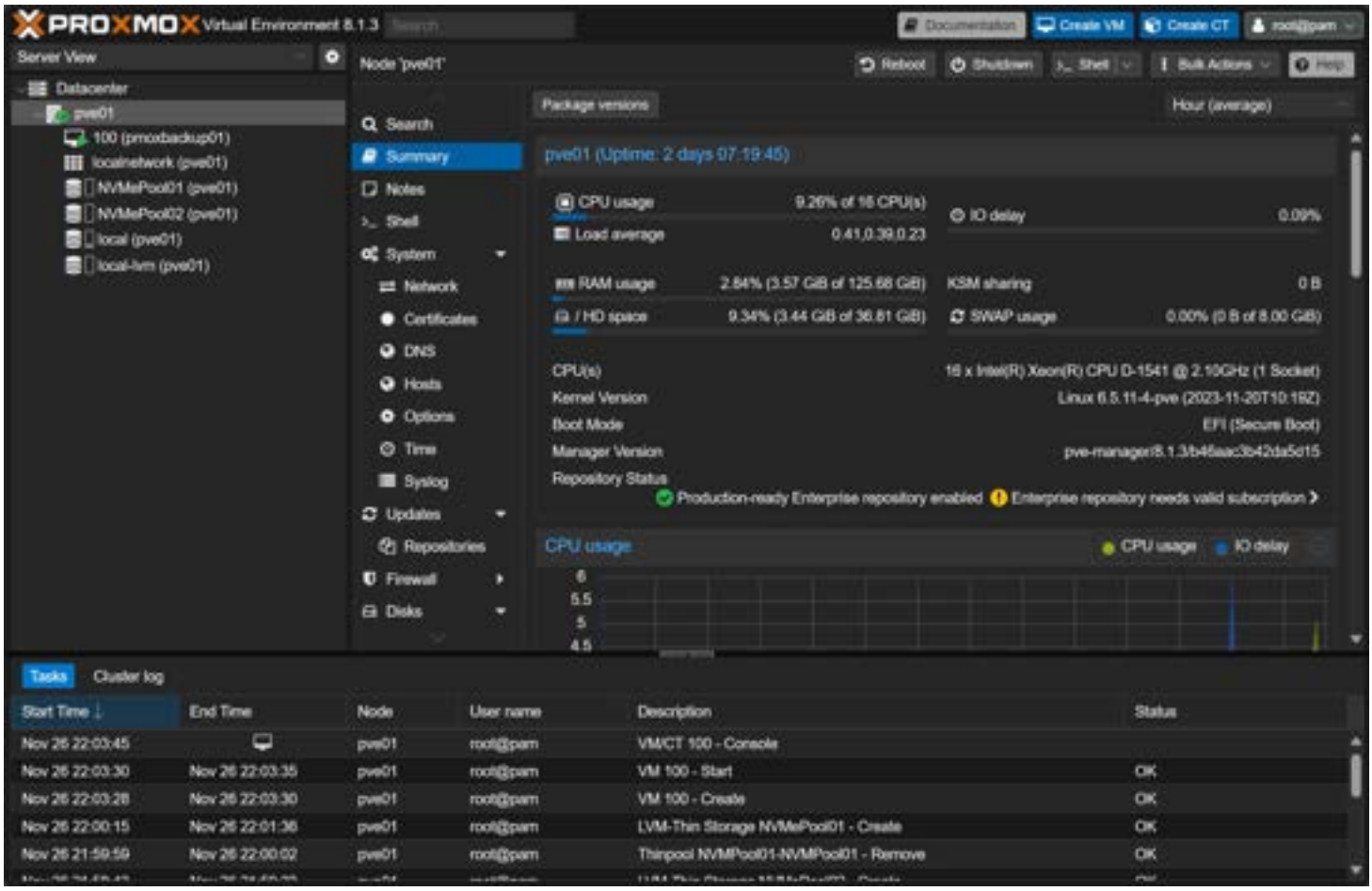
2023 11 26 22 06 06

Review the summary screen.



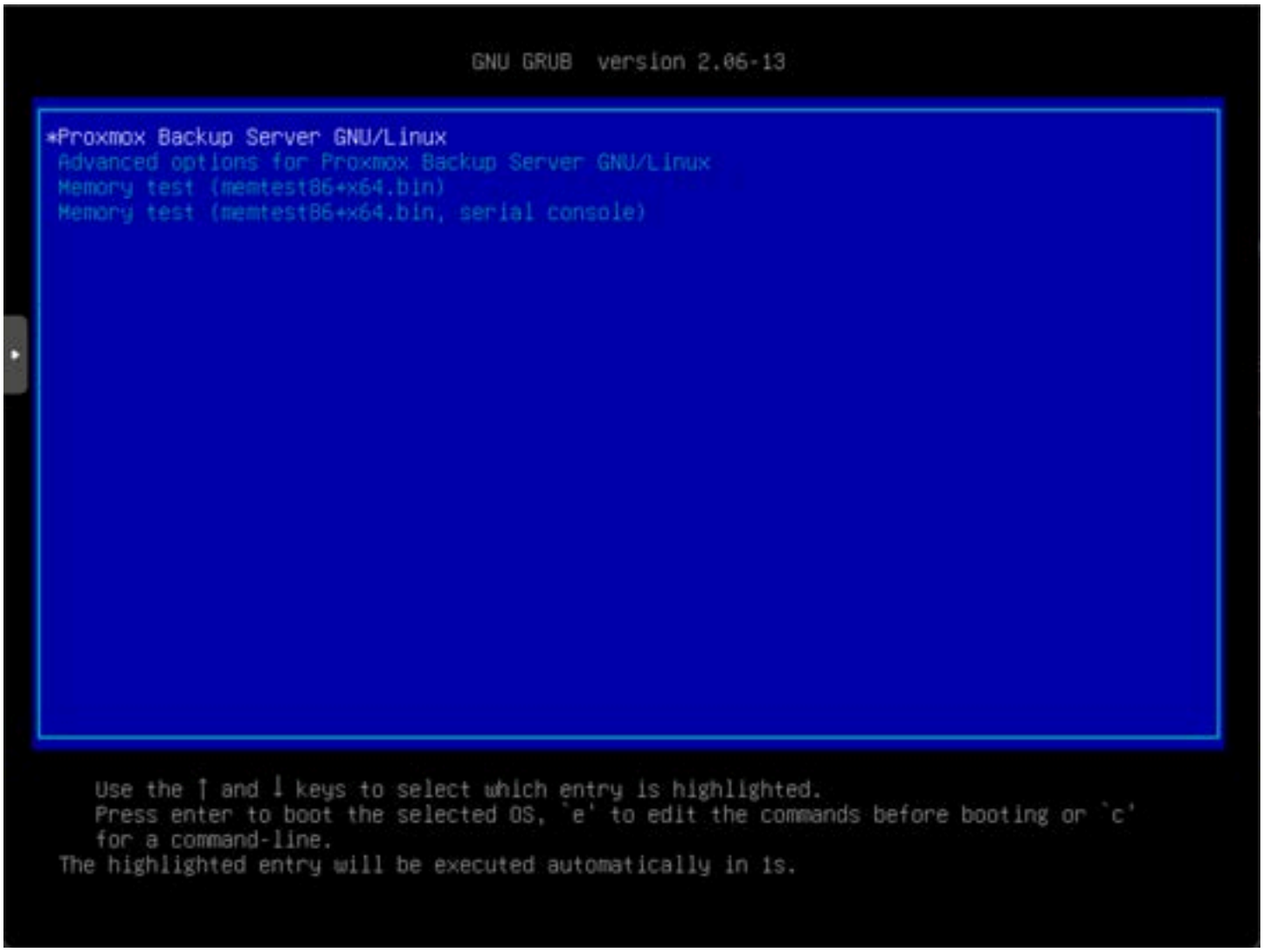
Summary screen for the installation process with PBS

Below is a screenshot of the Proxmox Backup Server running on Proxmox VE.



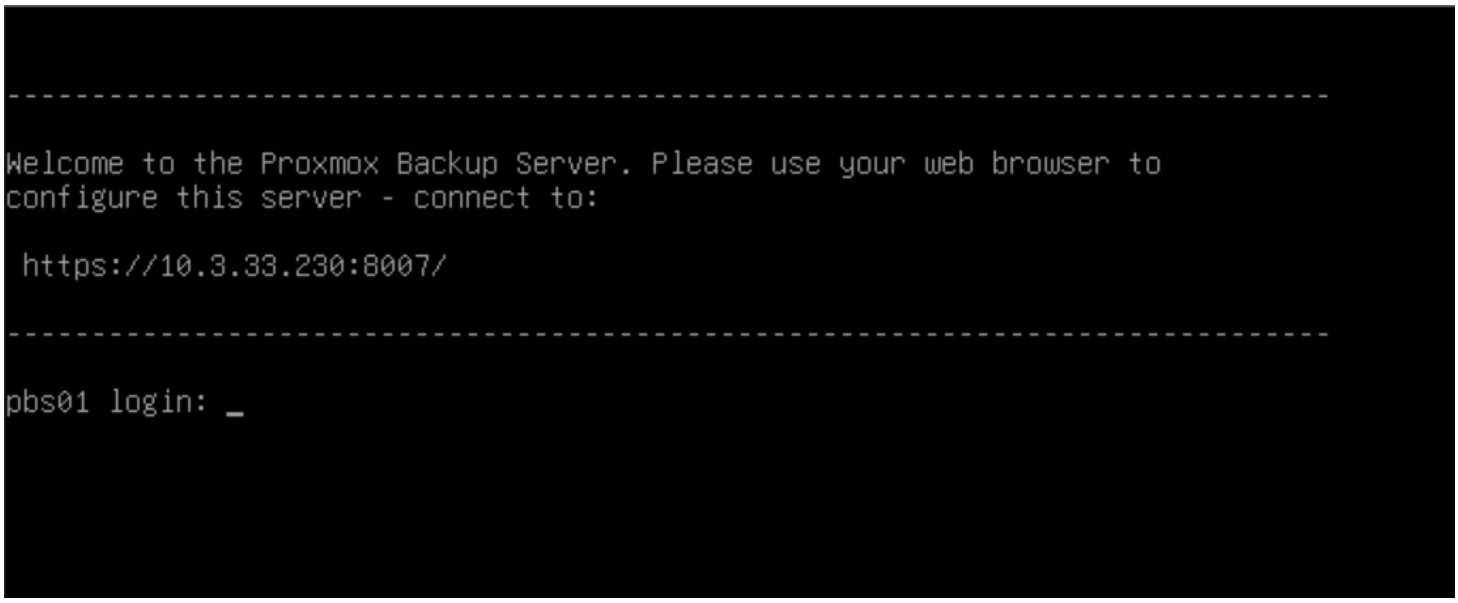
Running the proxmox backup server on proxmox ve

Booting the Proxmox Backup Server after installation.



Booting the proxmox backup server after installation

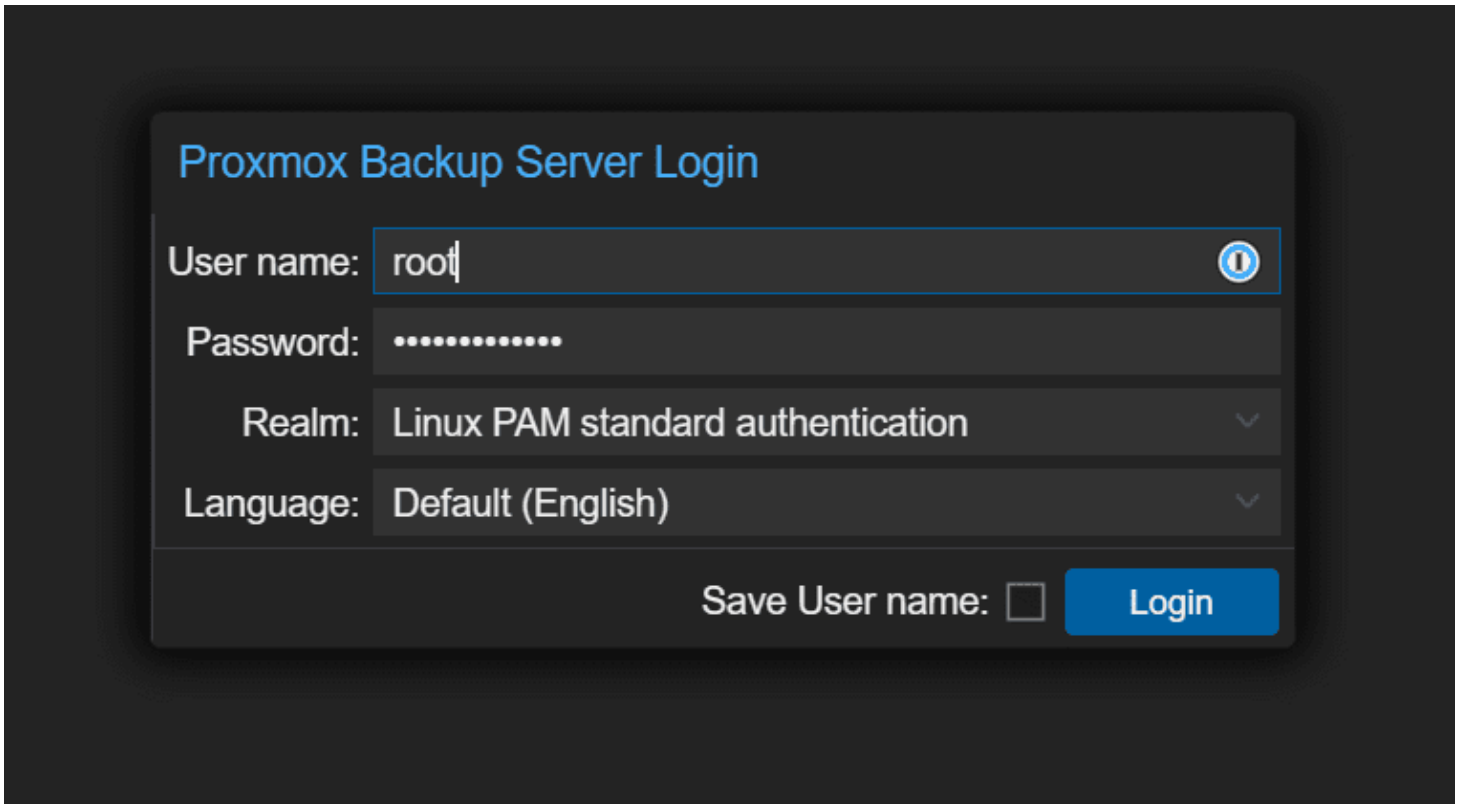
After the Proxmox Backup Server boots, you will see the default text splash screen directing you to open a browser to start managing the server. Note the port 8007, which is different from the Proxmox VE port 8006 for accessing the GUI. This is the command line interface (CLI) from the console.



Proxmox backup server login

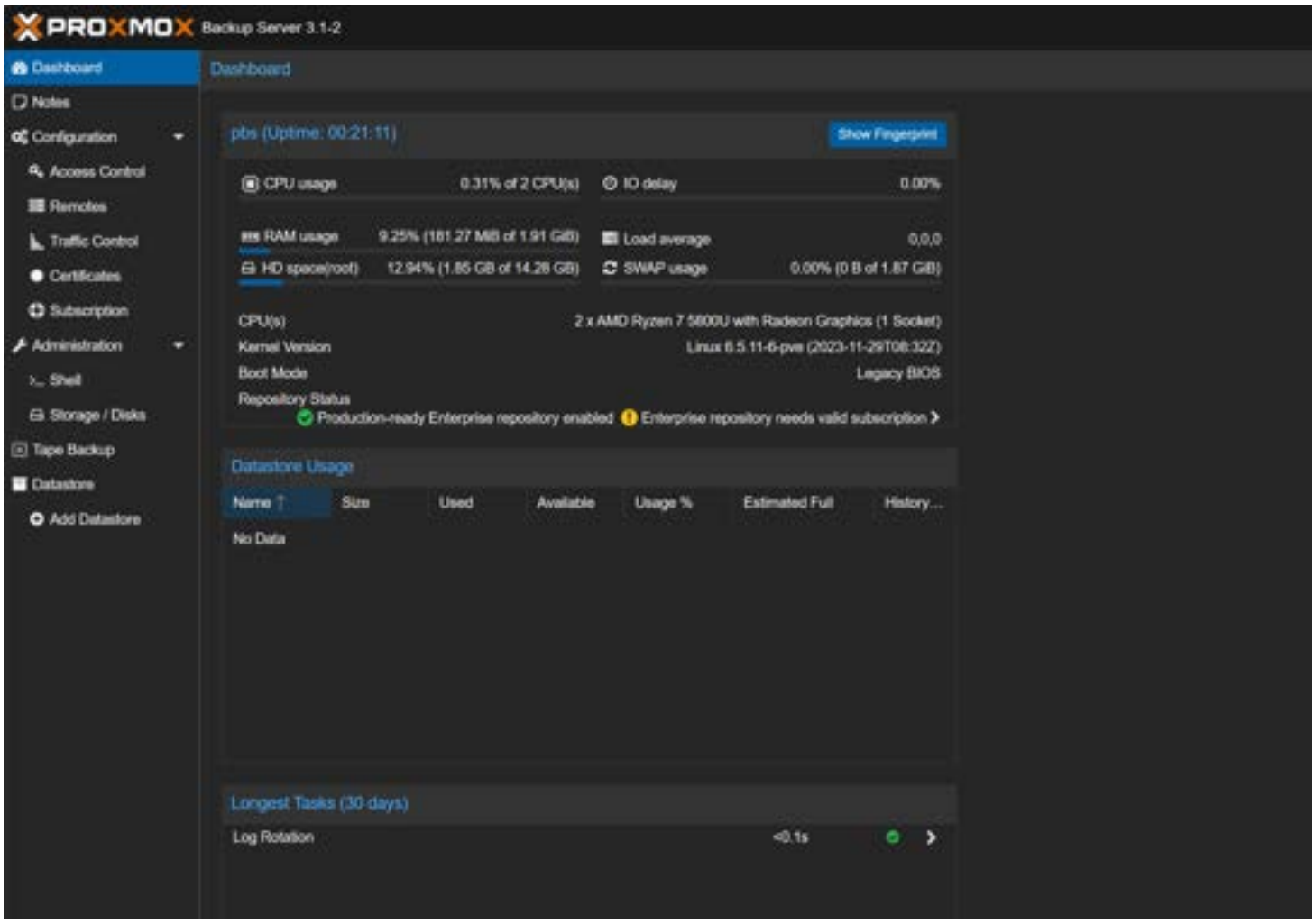
Logging into the Proxmox Backup client interface

Next, let's navigate to the web UI and log in to the web UI for the Proxmox Backup Server.



Pbs web interface login

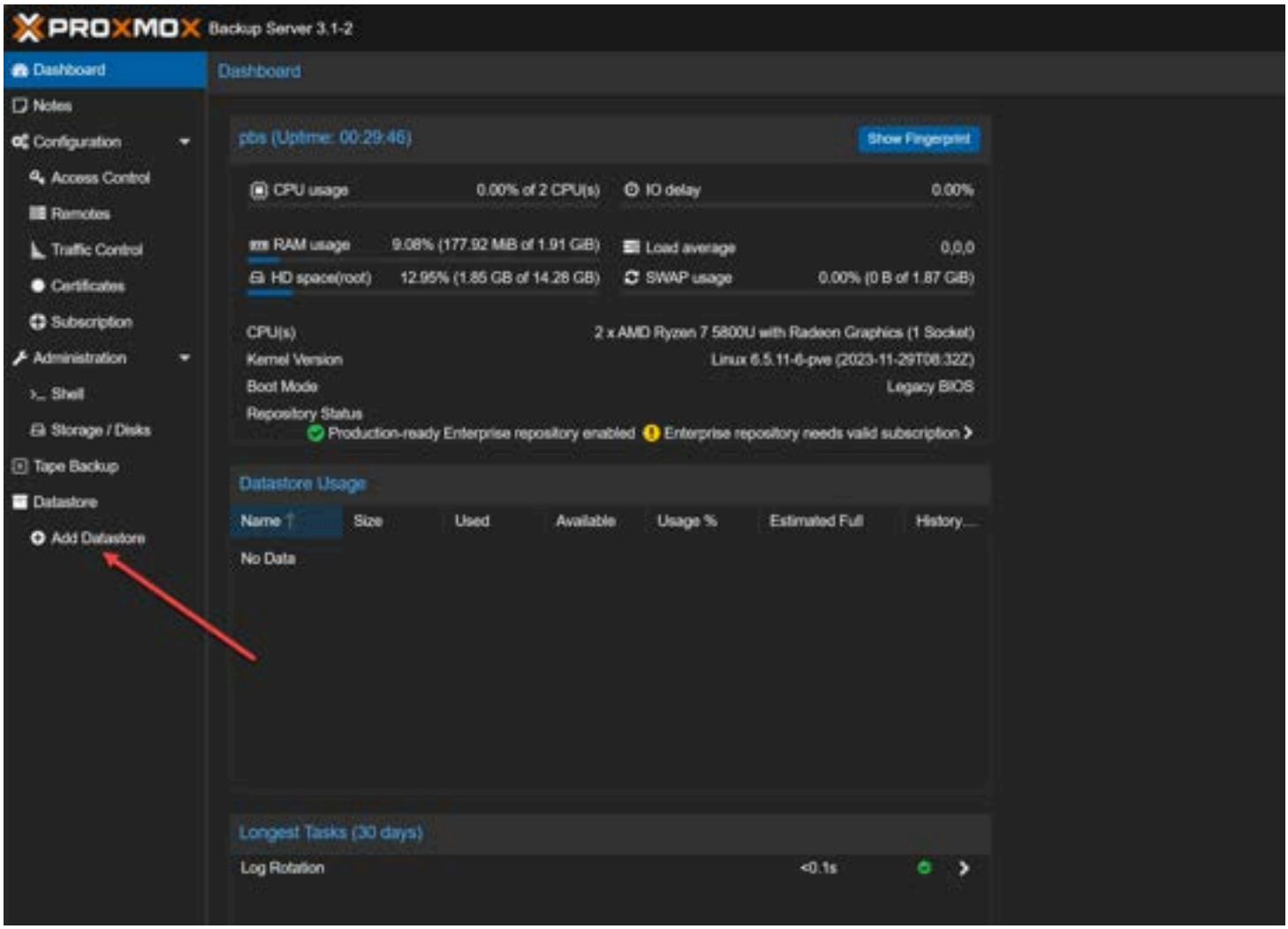
Below, we see the overview of the PBS tool.



Logged into PBS seeing the dashboard

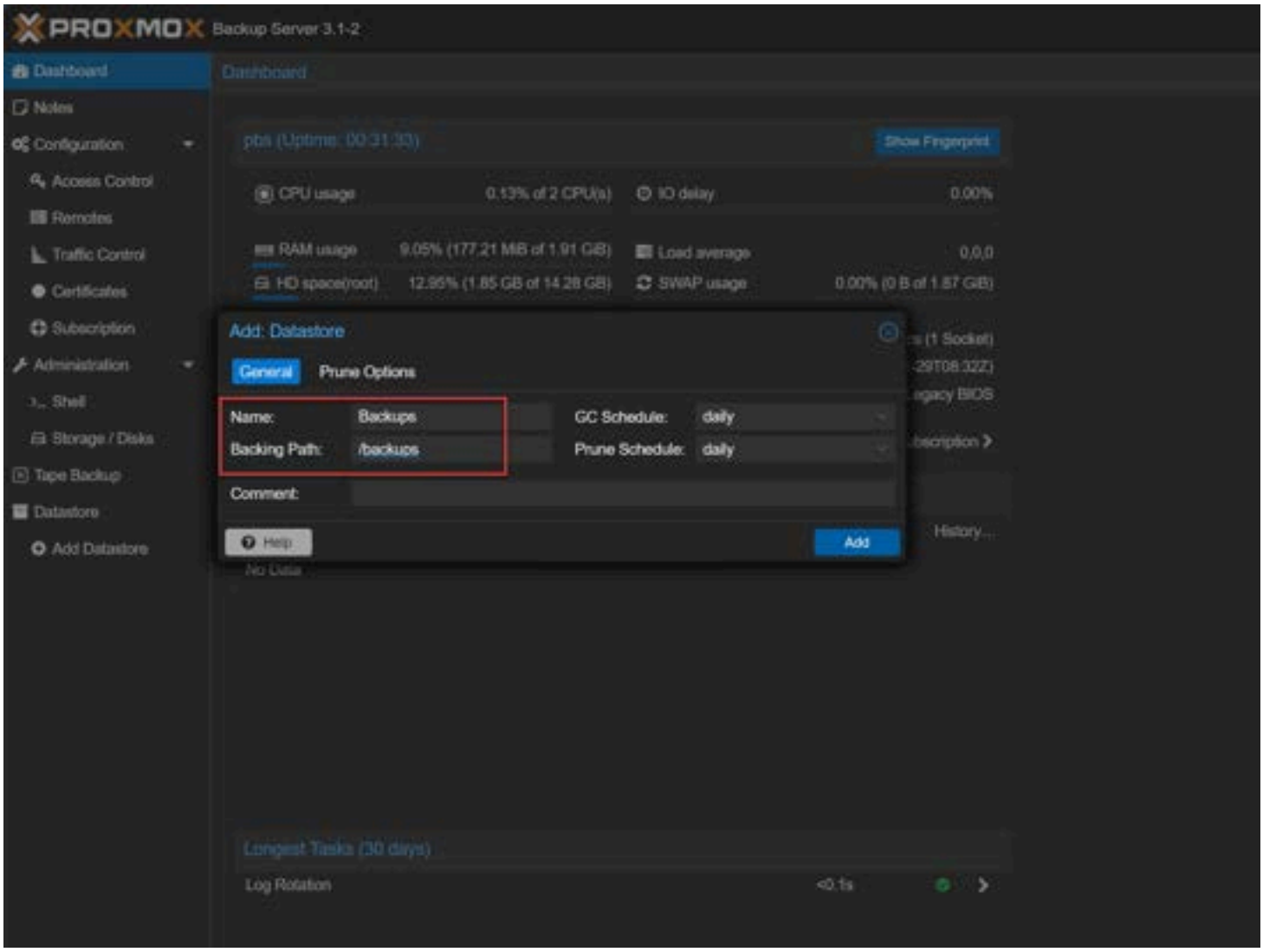
Adding a datastore for storing backups

Next, let's add a datastore for storing Proxmox backups. After logging into the Proxmox Backup Server, click the **Add Datastore** option under **Datastore**.



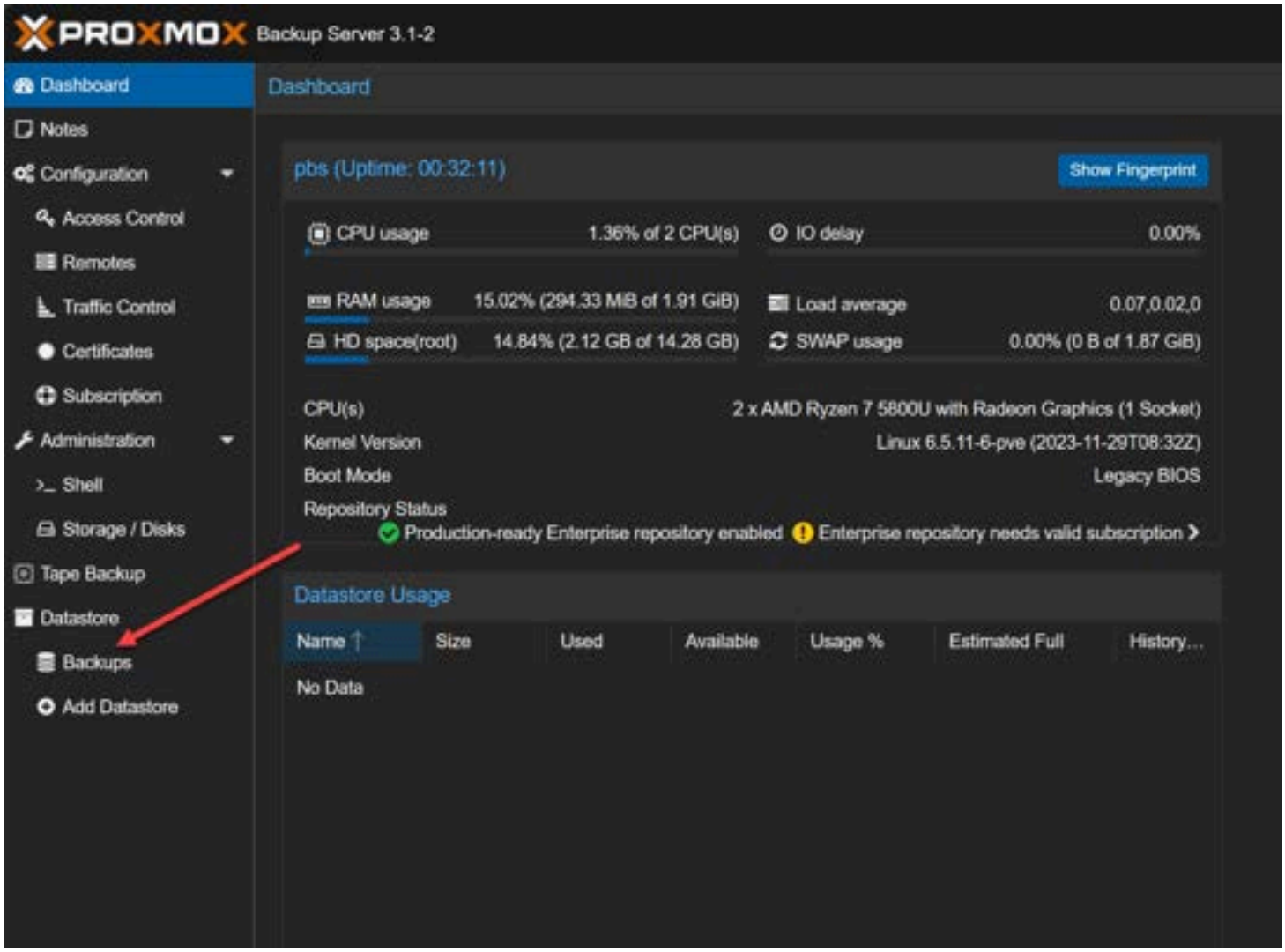
Beginning the process to add a datastore to the PBS

This will launch the **Add Datastore** dialog box. Name the new datastore and then enter the backing path. You don't have to create the backing path location as the process will do this for you. Click **Add**. This will add the backup storage location to *local storage* on your Proxmox Backup Server.



Naming the datastore and setting the path

Now, we see the datastore displaying in our Proxmox Backup Server.

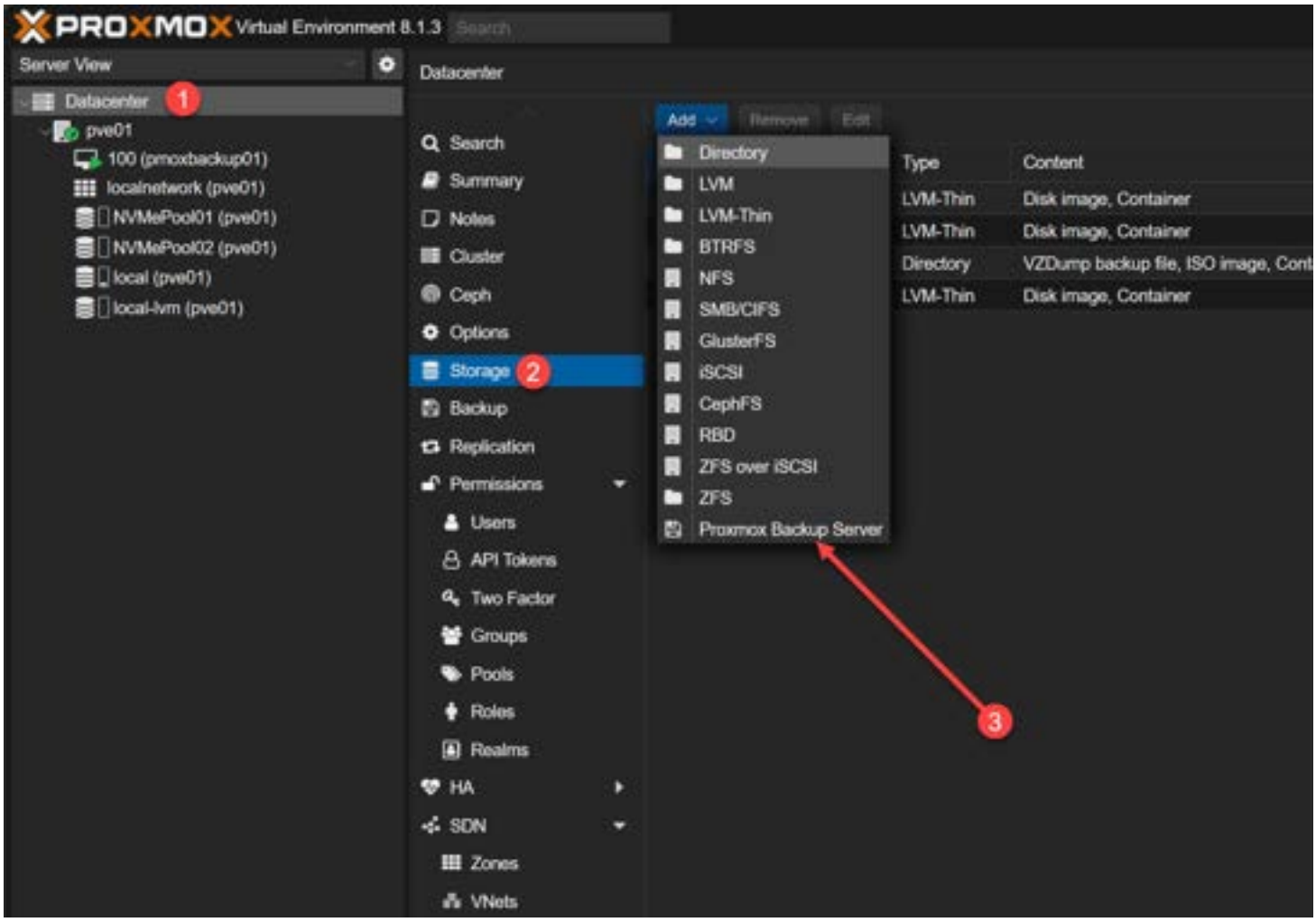


Viewing the datastore in PBS

Add the Proxmox Backup Server instance to your Proxmox VE server

On our Proxmox VE host, the Proxmox Backup Server is viewed as **Storage**. So, we first need to add it as storage to our Proxmox VE host.

Navigate to **Datacenter > Storage > Add > Proxmox Backup Server**.

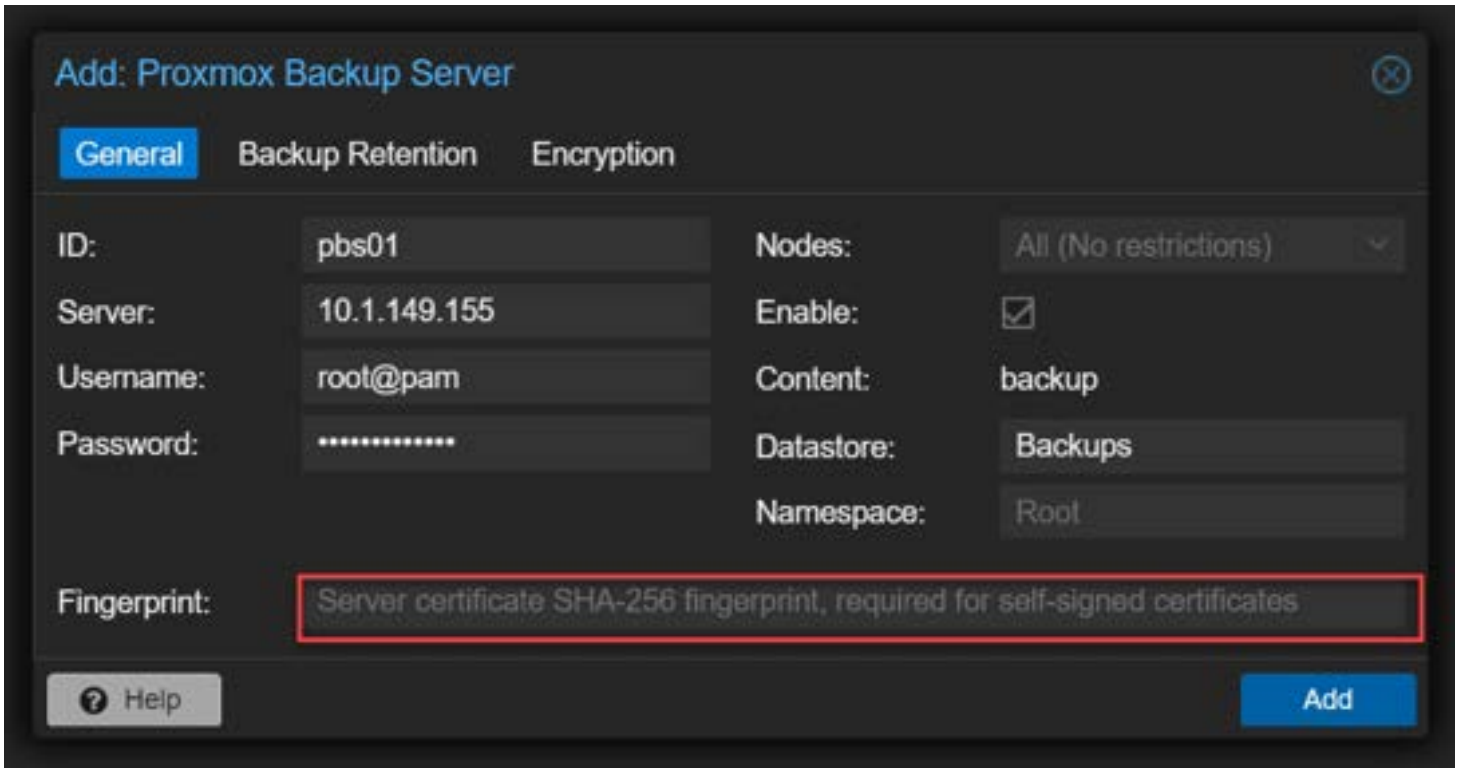


Adding PBS storage to your proxmox ve instance

This will launch the **Add: Proxmox Backup Server** dialog box. Here we enter the following information:

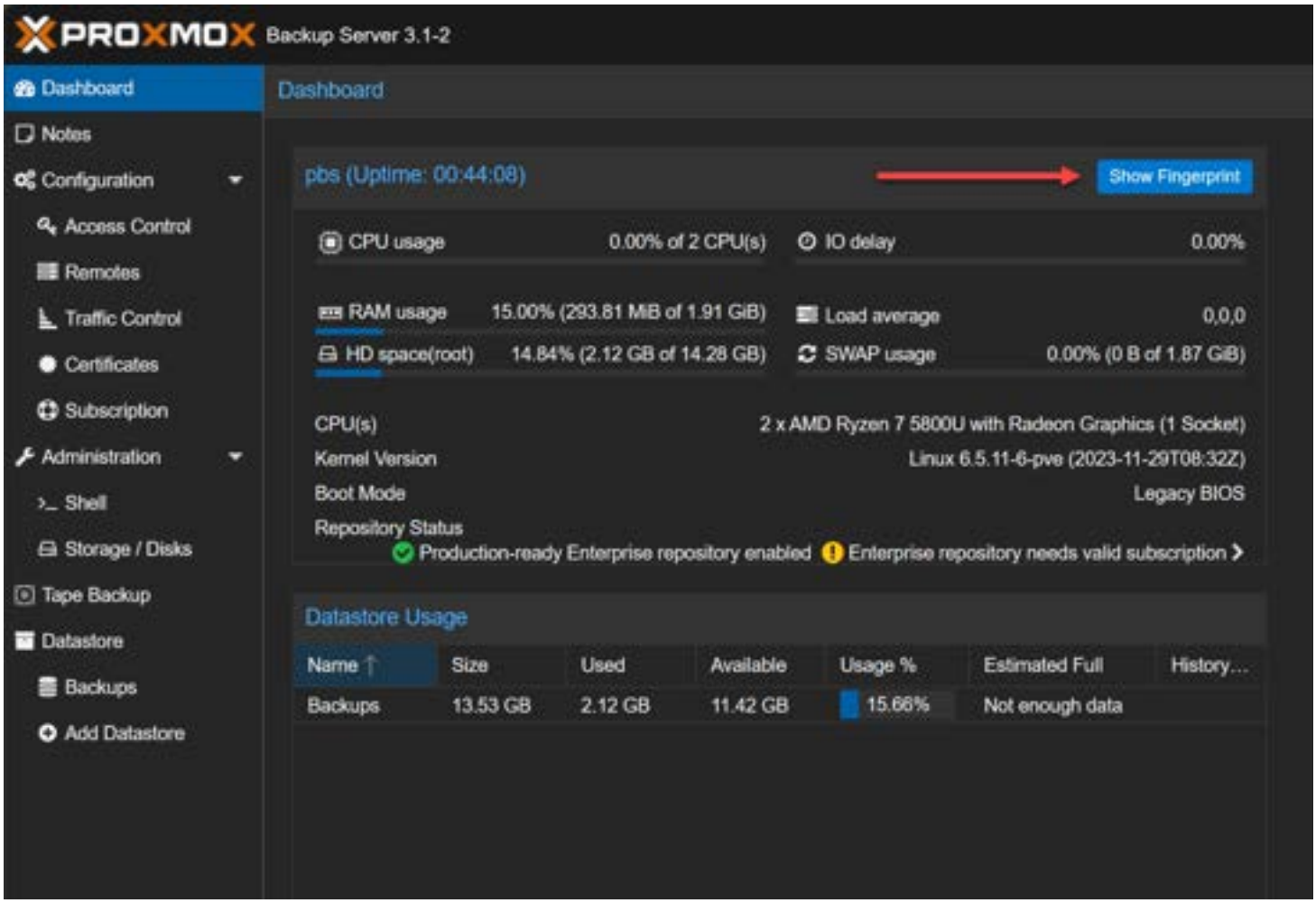
- ID
- Server
- Username
- Password
- Datastore
- Fingerprint

You may wonder where we get the fingerprint.



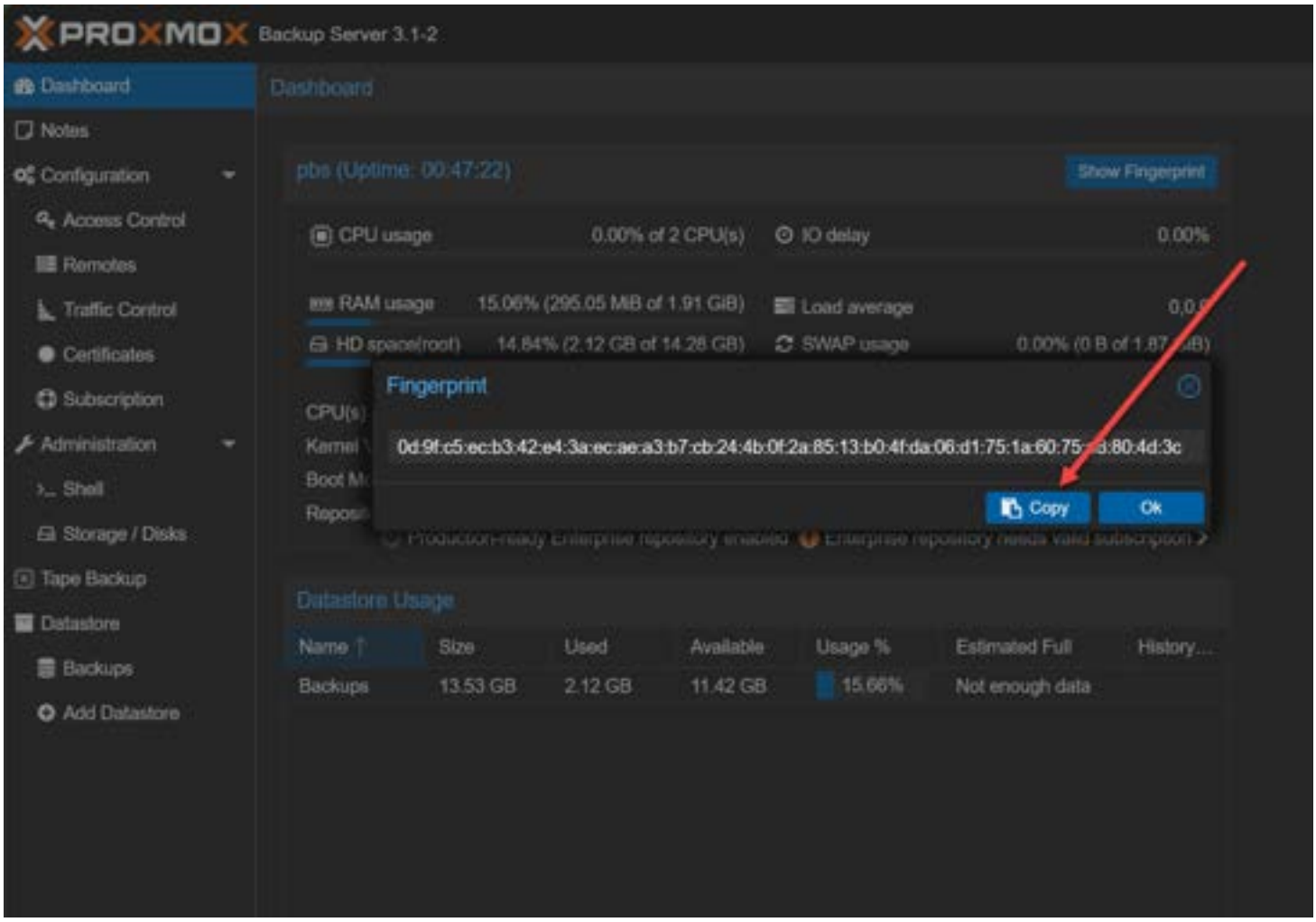
Add the PBS certificate thumbprint

Navigate back to your Proxmox Backup Server and click on the **Dashboard > Show Fingerprint** button.



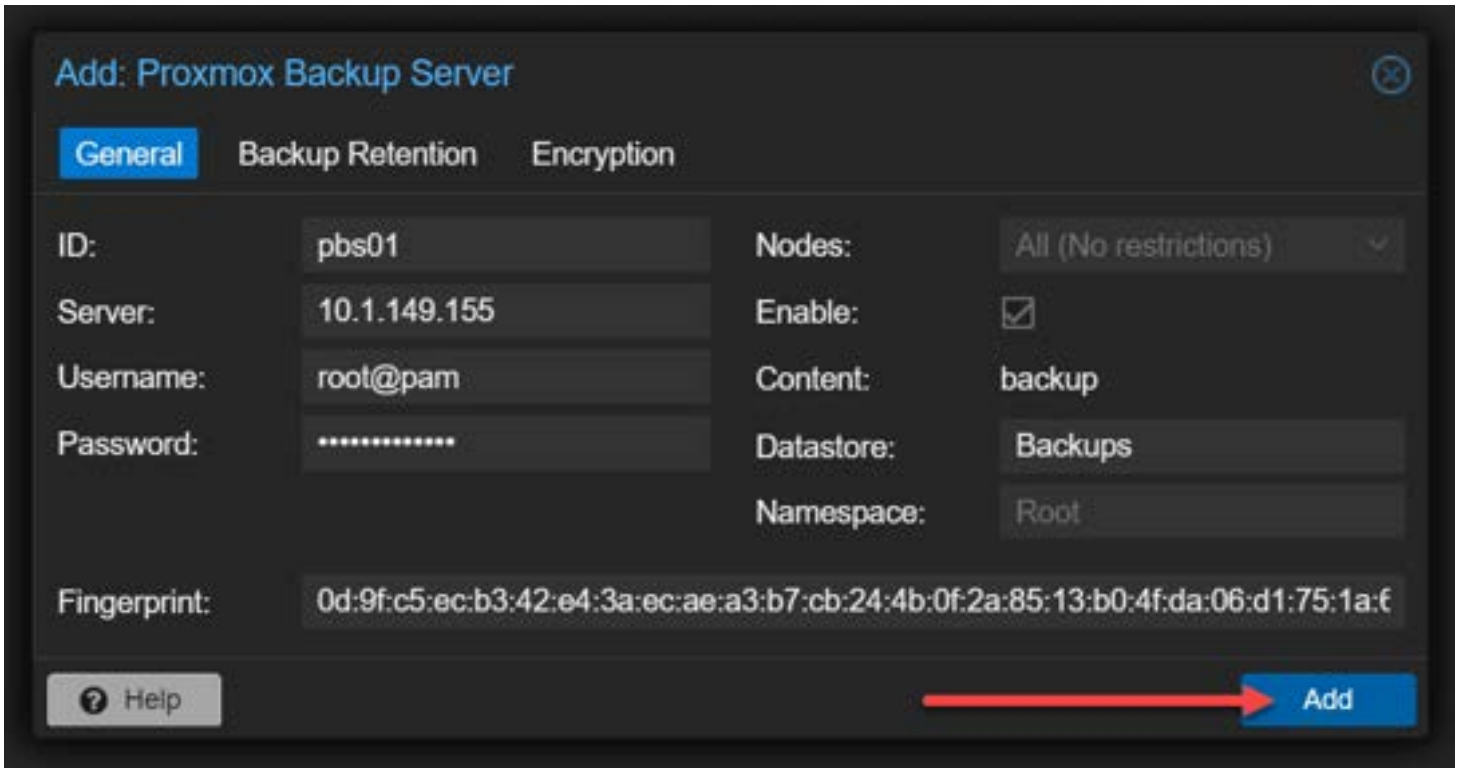
Show fingerprint on PBS

Here, we can copy the fingerprint.



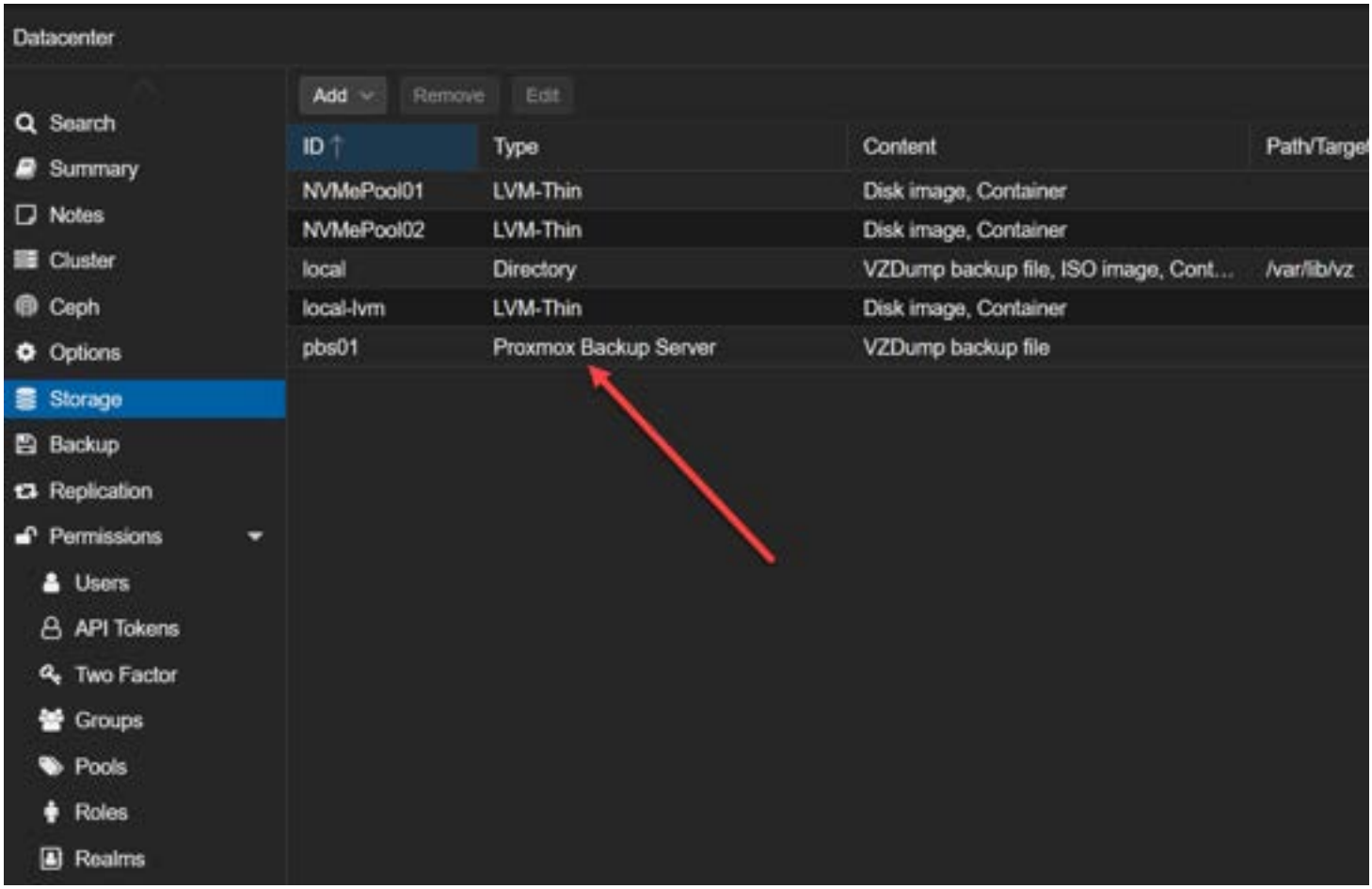
Copy the fingerprint of the PBS server

Now, we can paste in the fingerprint.



Thumbprint added and ready to save to connect to pbs

After adding the fingerprint and clicking the **Add** button, we see the Proxmox Backup Server listed.



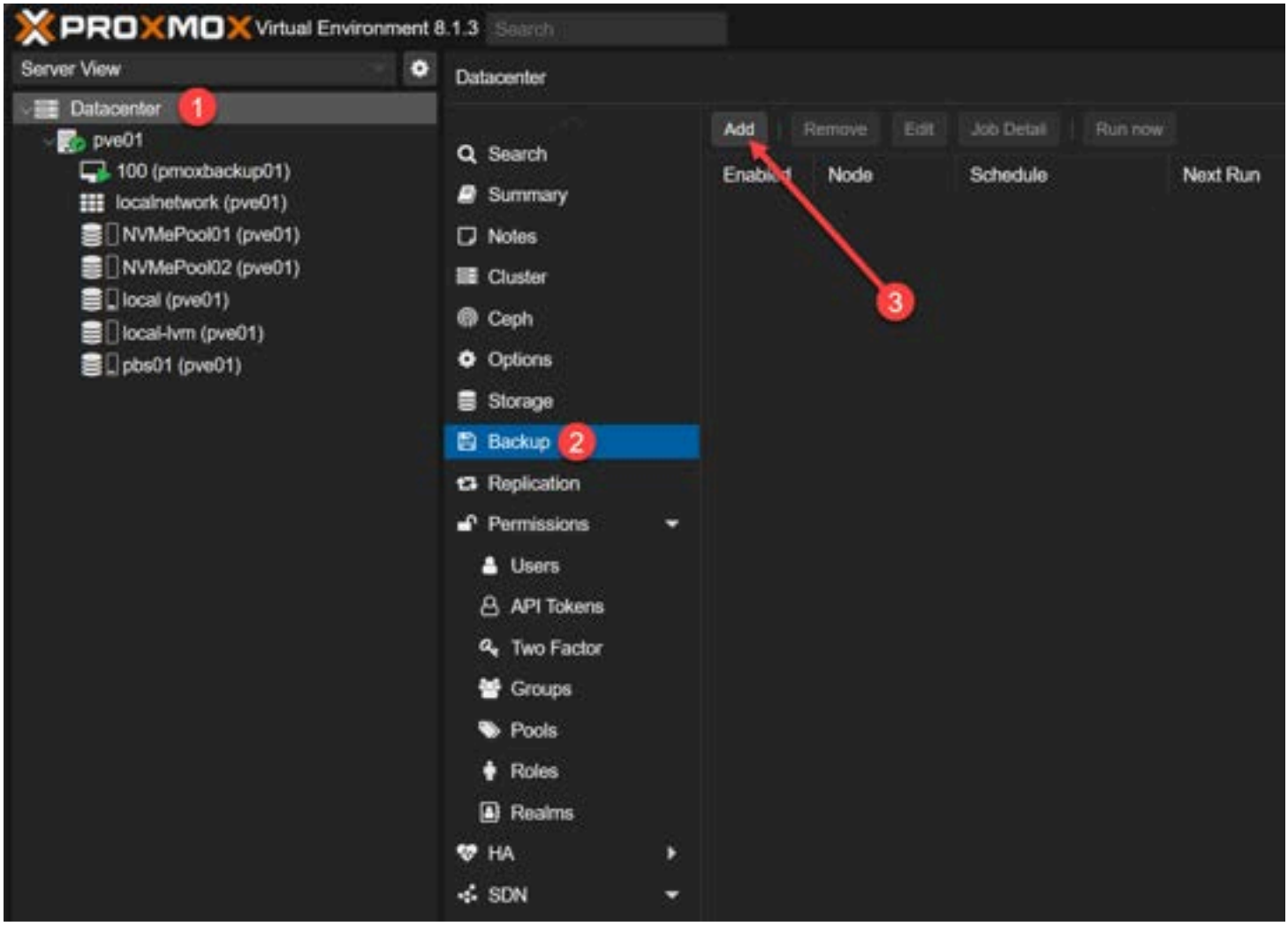
Proxmox backup server storage added successfully

Creating a backup job

Now that we have the Proxmox Backup Server added as storage to our Proxmox VE host, let's create a backup job.

You might think we would do this from the Proxmox Backup Server side. However, we create the Proxmox backup job from the Proxmox VE host. Proxmox Backup Server offers advanced features like snapshot mode and customizable backup retention policies.

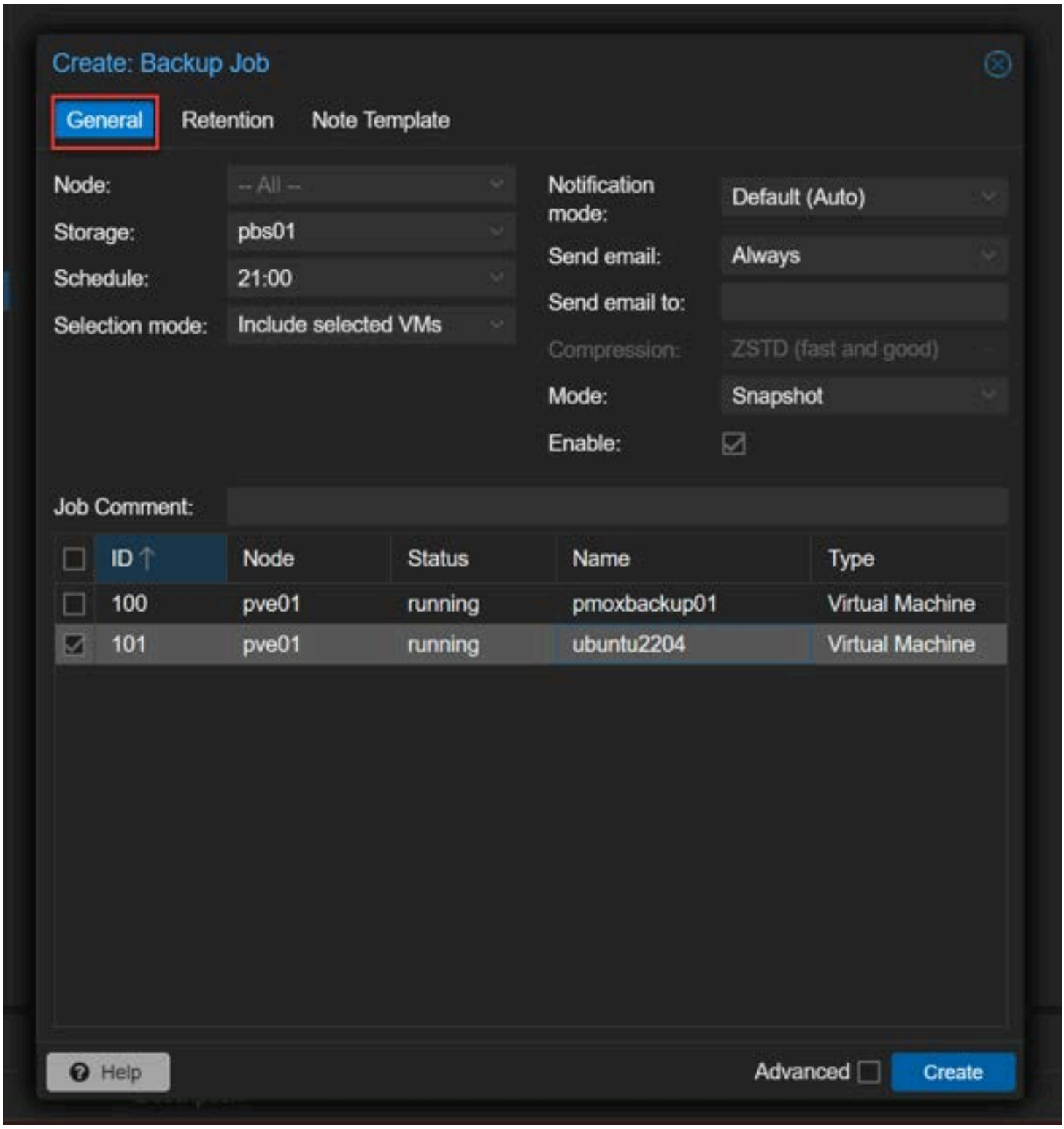
Navigate to **Datacenter > Backup > Add**.



Beginning to add a new proxmox backup job

On the general tab, we can set all the main backup option selection for our Proxmox backup job. This includes:

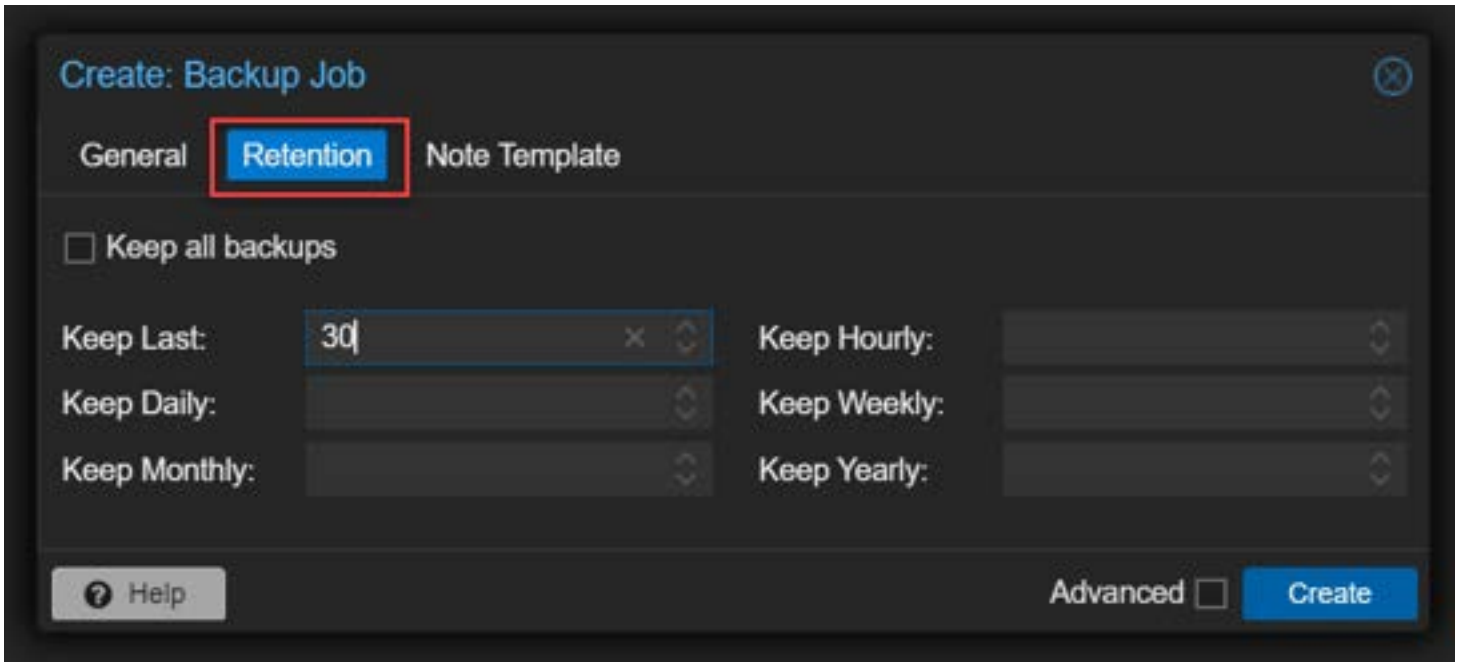
- Source nodes from which to backup
- Storage that we want to target for the backup
- Schedule when your backup job will runReading the previous sentence, it becomes evident that determining the storage and backup schedule are crucial aspects when setting up a Proxmox backup server.
- Selection mode (which VMs)
- Notification mode
- Send email options
- Send email to for configuring an email address
- Backup mode – snapshot, suspend, stop



The general tab on the create backup job box

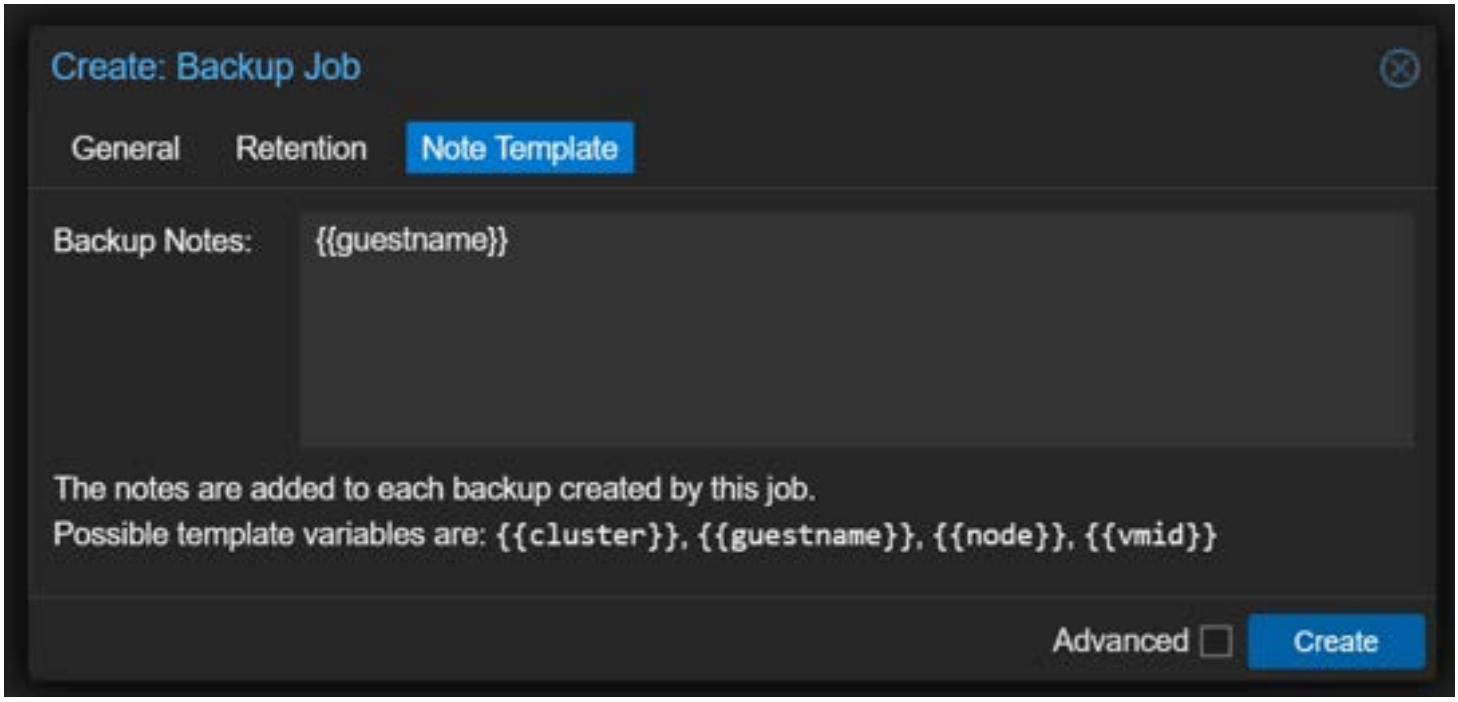
On the retention screen, you can configure all things retention and archive related, including:

- Keep all backups
- Keep last
- Keep daily
- Keep monthly
- Keep hourly
- Keep yearly



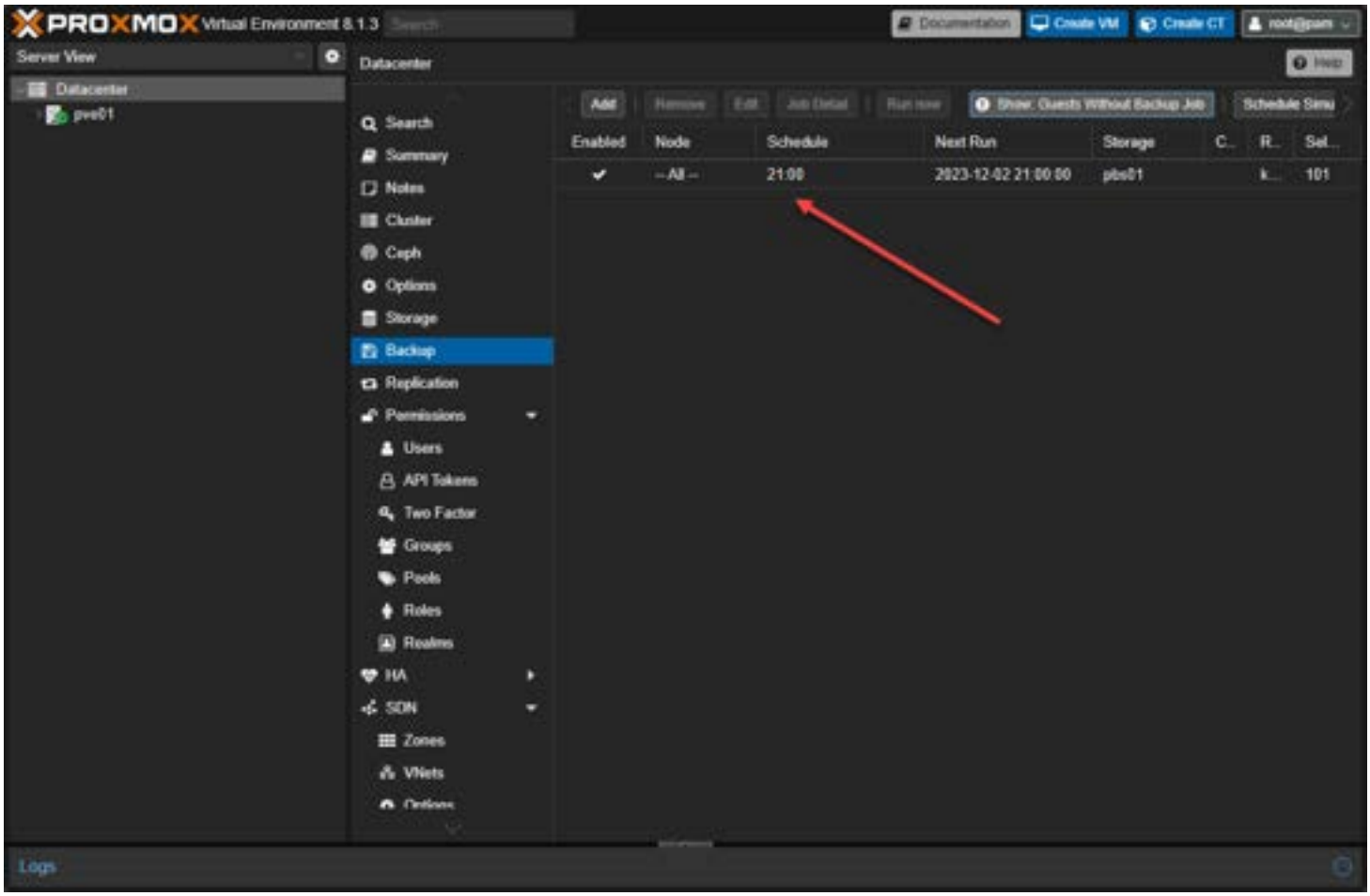
The retention tab on the new proxmox backup job box

You can also configure the **Note template**.



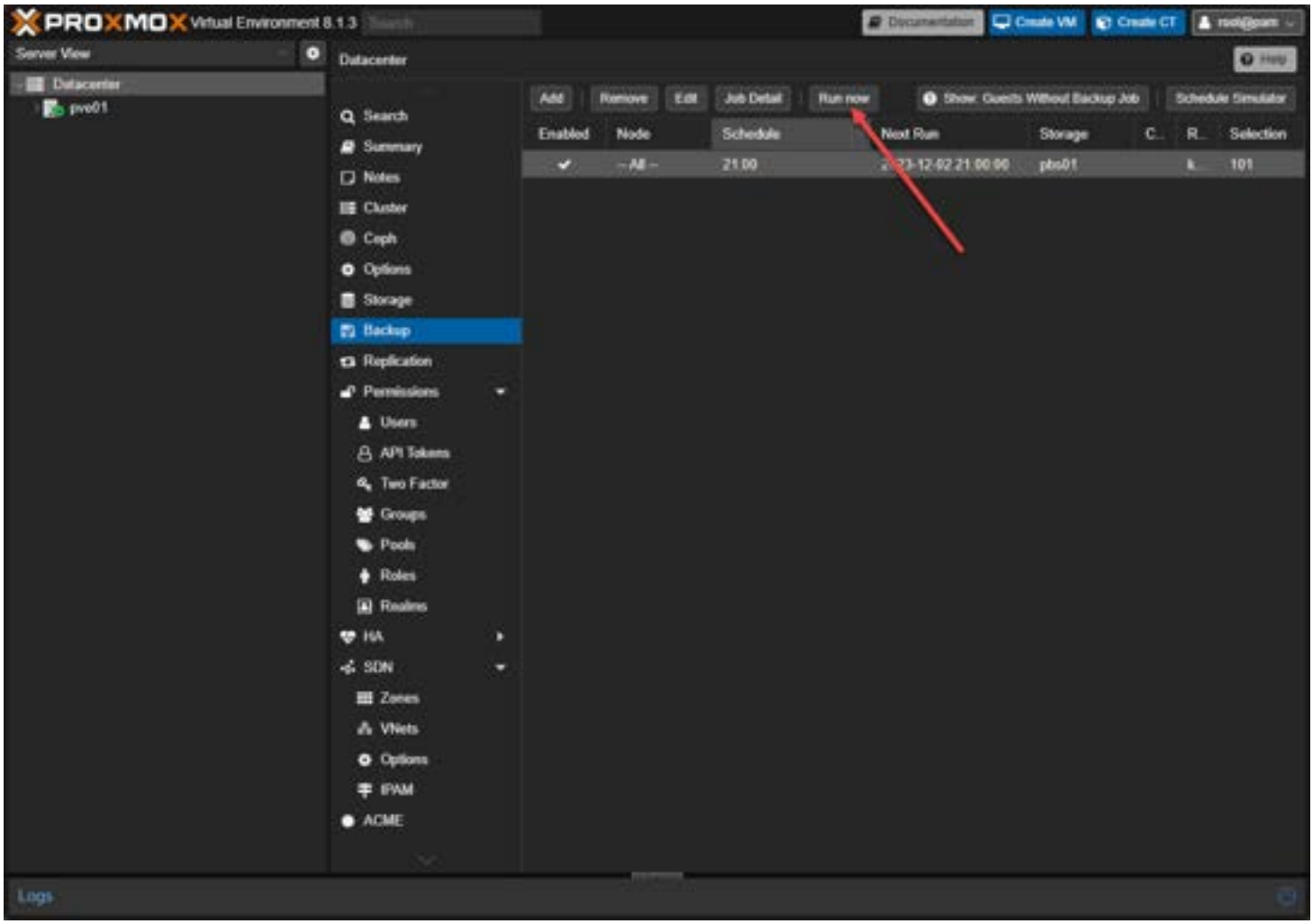
The note template for the new proxmox backup job

After creating the backup job.



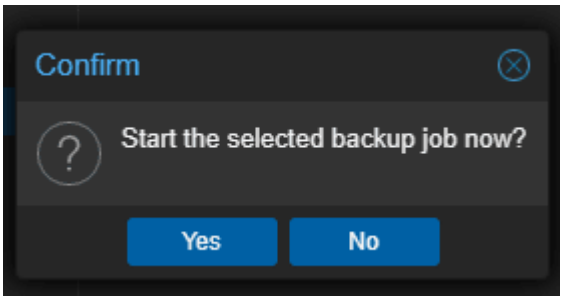
The newly created backup job proxmox virtual machine

We can choose to run the backup job now:



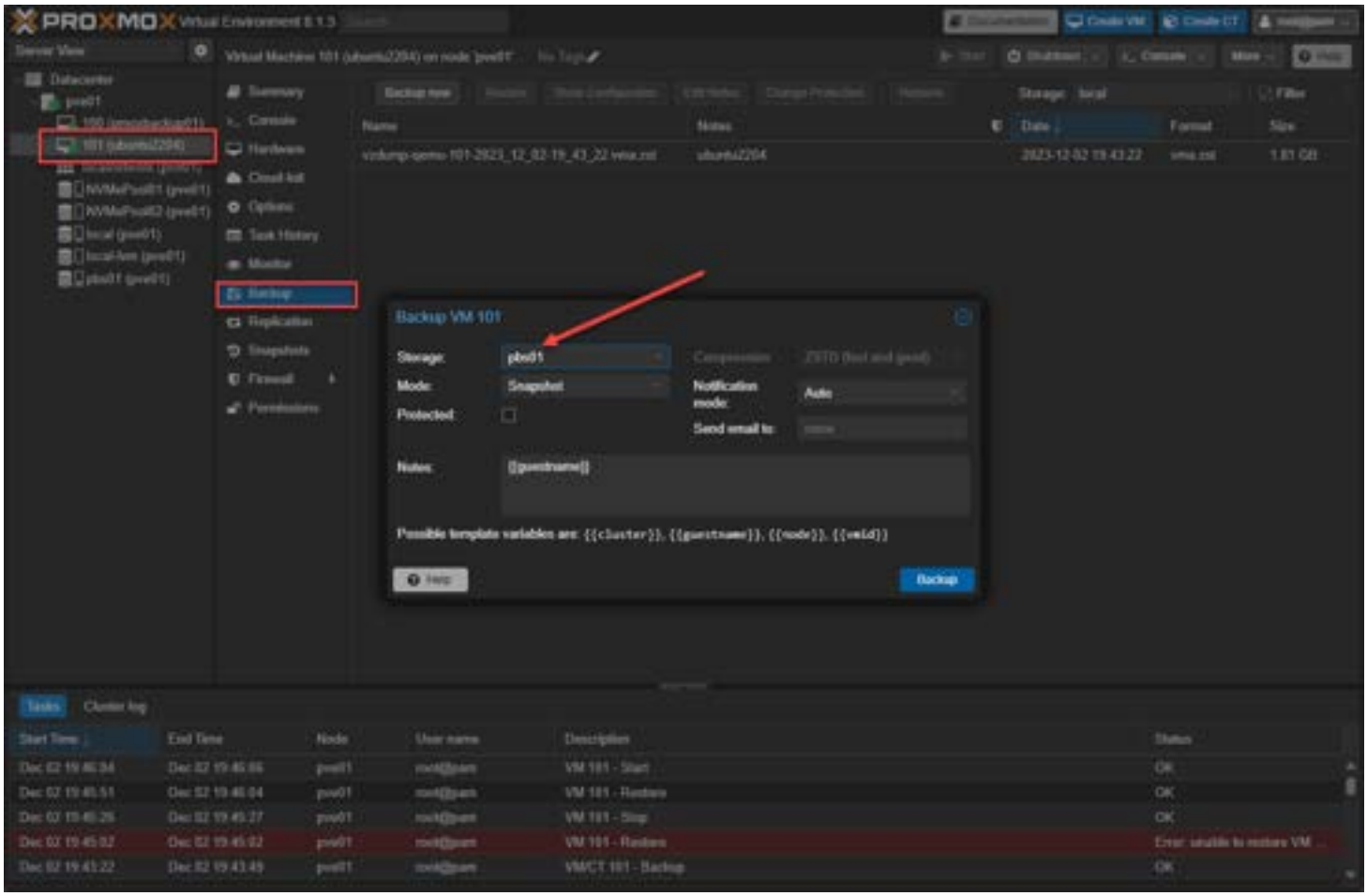
Choosing to run the new proxmox backup job now

Confirm you want to run the backup now.



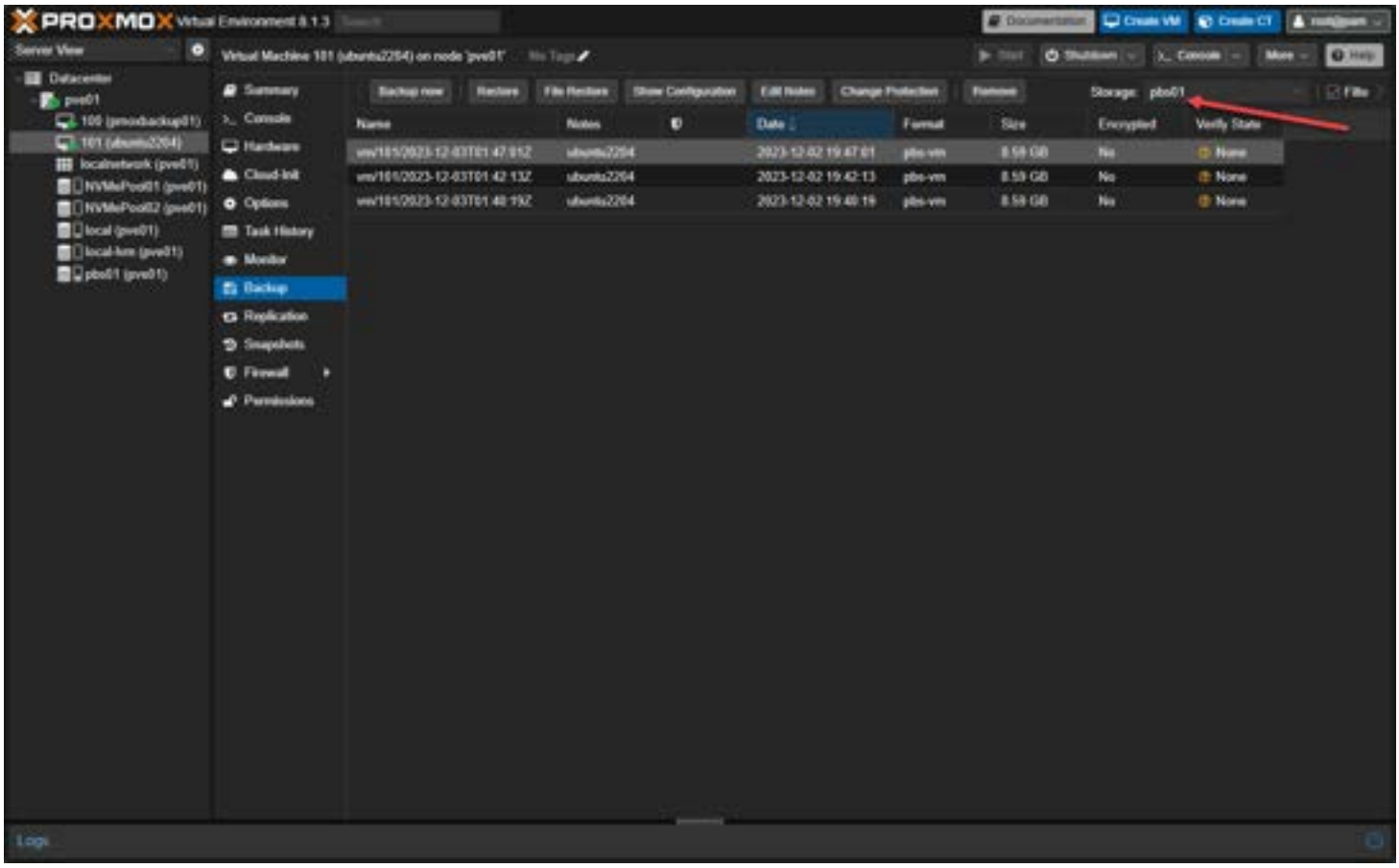
Confirm you want to run the new proxmox backup job

You will also note, that you can navigate to the Proxmox VE host virtual machine and select **Backup** and you will have the option to backup your VM from here. Make sure you choose your PBS storage location.



Confirm your backup storage

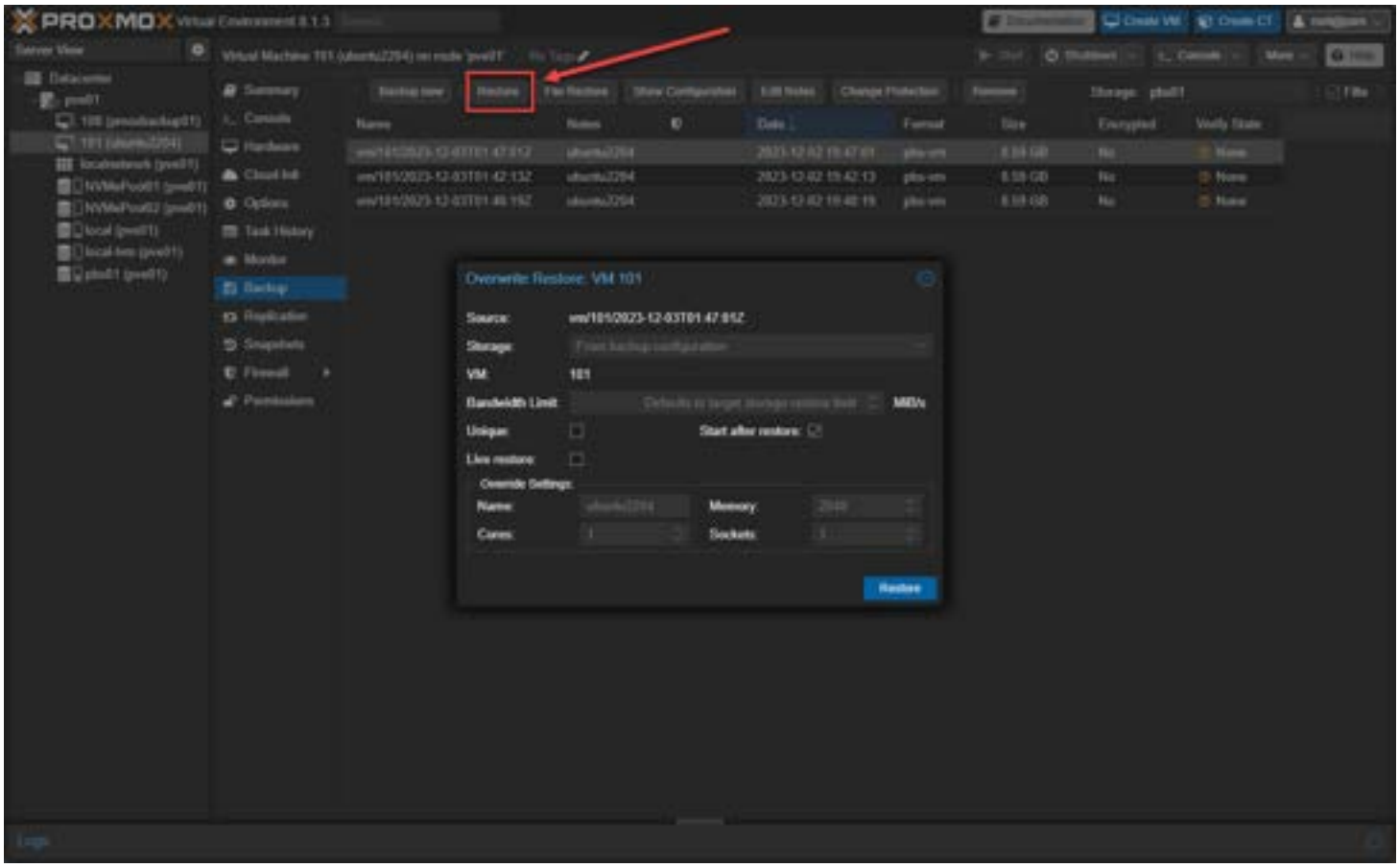
After you run the backup, you can see the vzdump backups details for the VM by selecting your PBS storage location.



Look at the backups on your remote PBS datastore

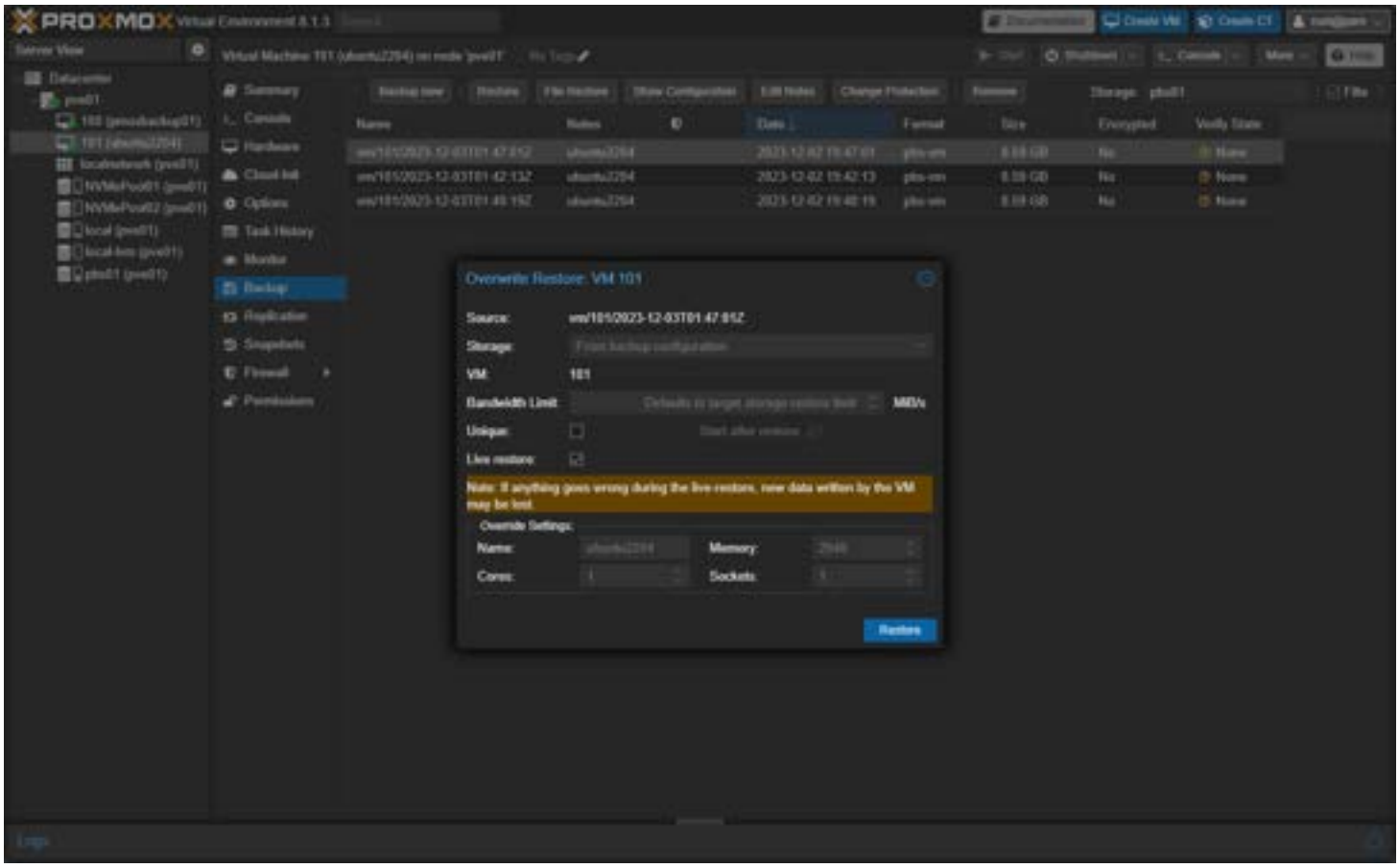
Restoring a Proxmox virtual machine from backup

After selecting your PBS backup location, you can select one of the restore points and select **Restore**



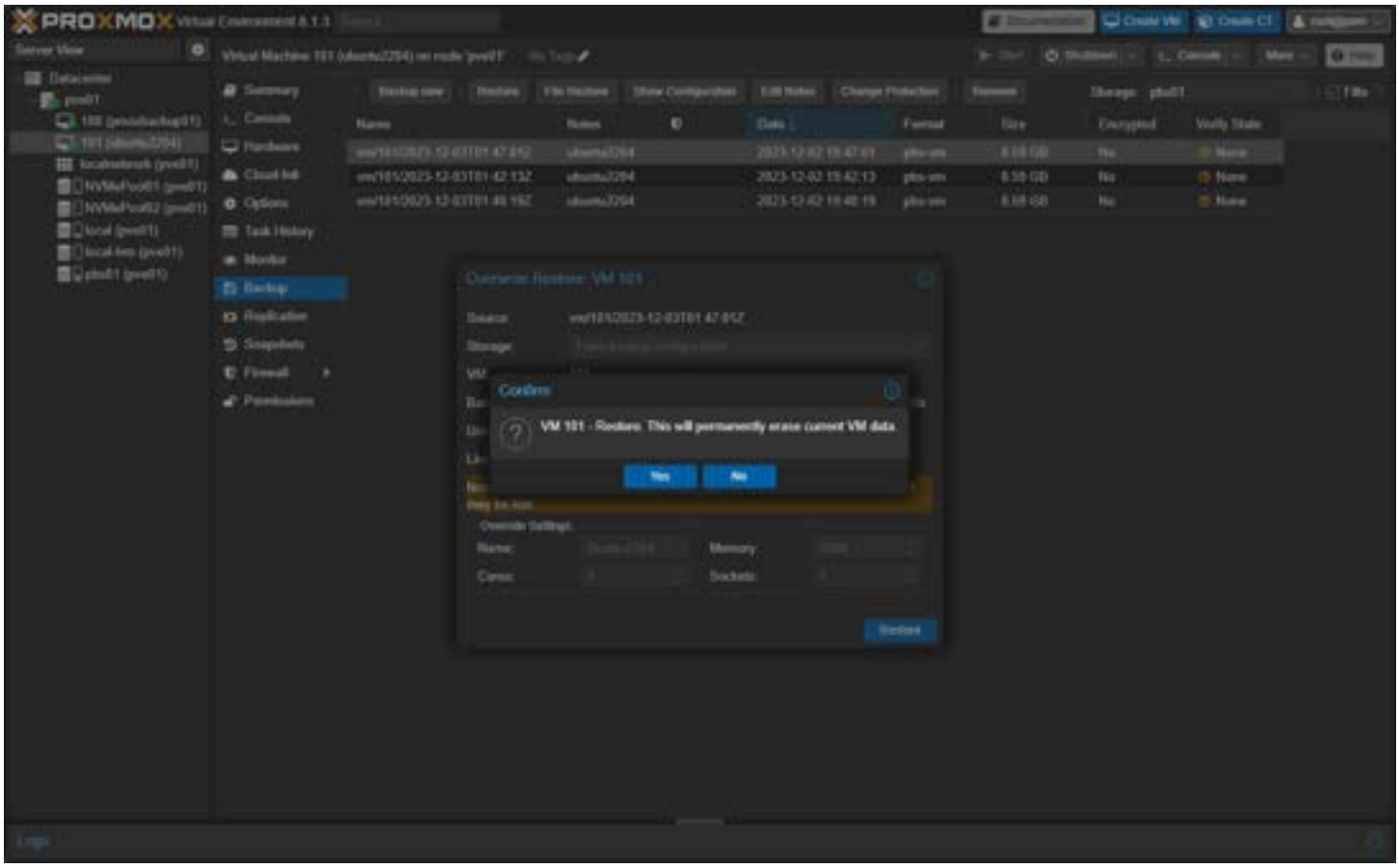
Beginning the restore process for a proxmox virtual machine

You can select to start the VM and also perform a Live restore.



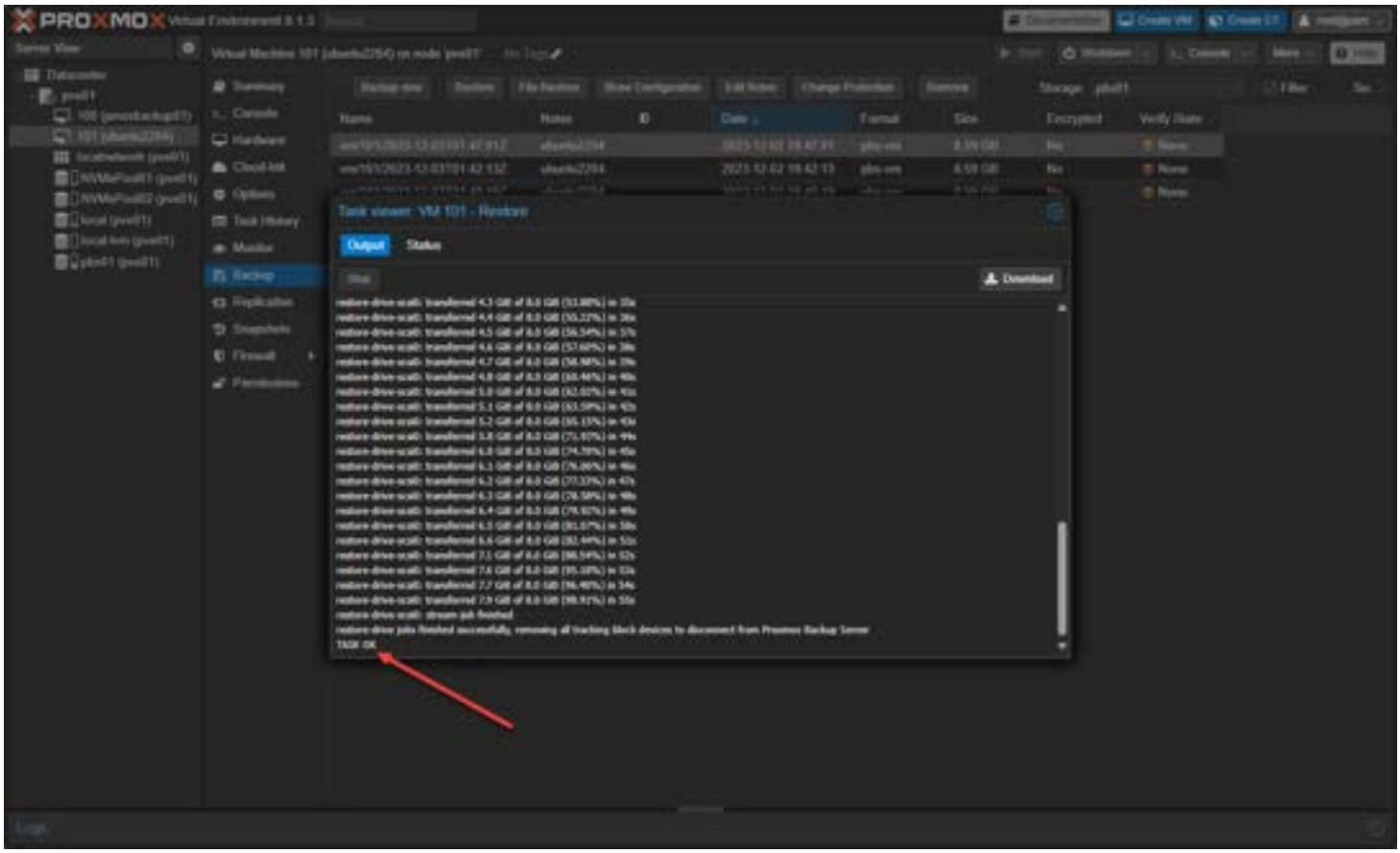
Choosing to overwrite with a live restore and power on the vm

Confirm you want to restore the VM.



Confirm you want to overwrite the vm

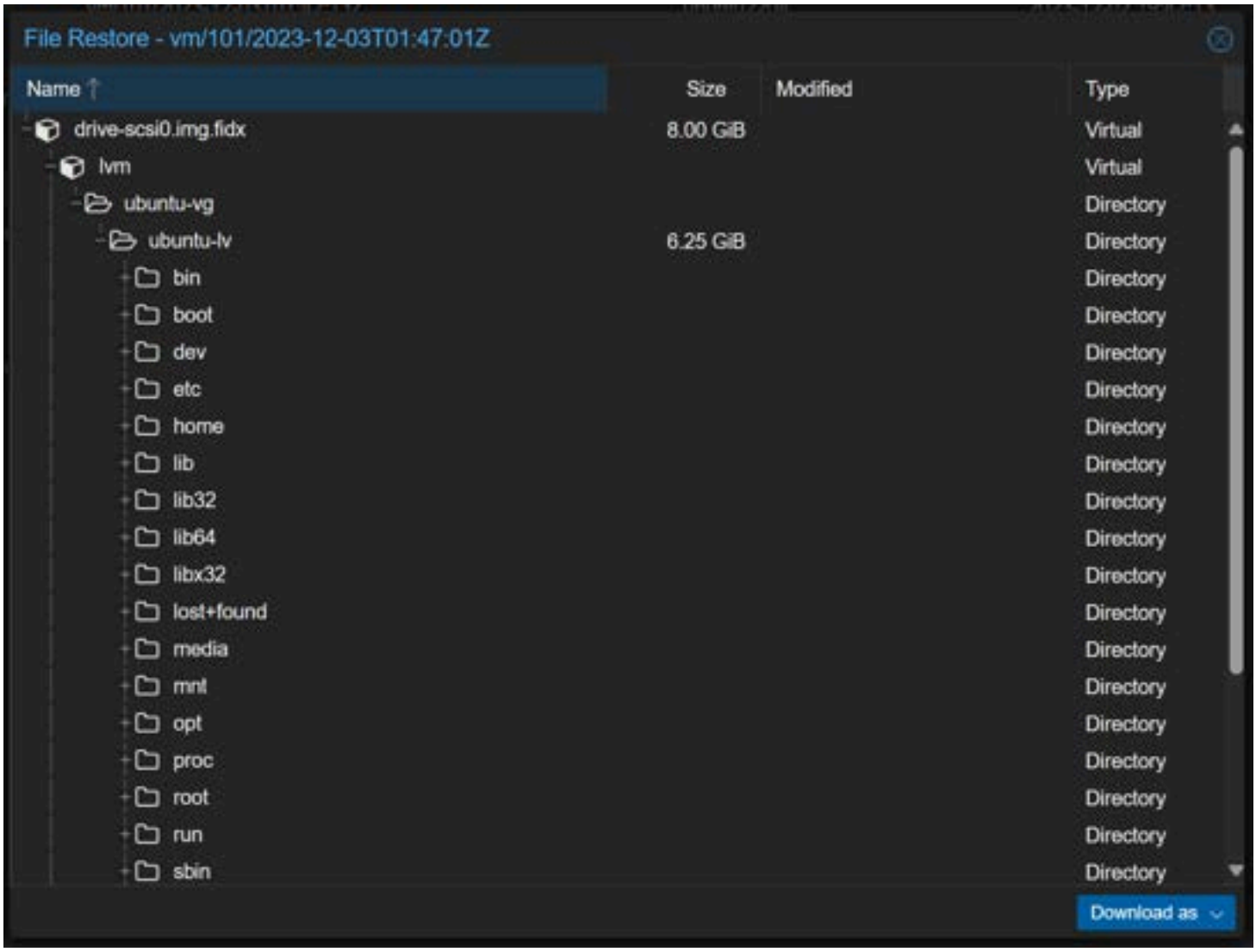
The task will progress and should finish successfully. The speed will depend on the network bandwidth you have between your Proxmox VE host and your Proxmox Backup Server.



The restore task completes successfully

Granular file restore

You can also click the **File Restore** button to restore individual files from the backups.



Running a file restore for a proxmox virtual machine

Frequently Asked Questions

How does PBS ensure the security of my backups?

Proxmox Backup Server improves data security with features like backup encryption and secure data transfer protocols. This means your backups are protected from unauthorized access and potential threats.

Can I use PBS for backing up physical servers as well as virtual environments?

Yes, it can handle backups for both virtual machines and physical hosts. However, it seems this functionality as documented from Proxmox is more on the roadmap for better integration and functionality. It is mentioned you can use the Proxmox Backup Server client to back up a physical host. Check the thread here: PBS: How to back up physical servers?

What are the advantages of using incremental backups in PBS?

These save time and storage space. By only backing up data that has changed since the last backup, the load on the network and storage is reduced.

Is it possible to automate backup jobs with Proxmox Backup Server?

Proxmox Backup Server allows you to automate backup jobs through its scheduling feature. This automation ensures regular backups without manual intervention, making backup management more efficient.

What storage options are available with Proxmox Backup Server?

You can configure ZFS pools and RAID configurations for internal storage. You can also configure external storage solutions like NFS and SMB/CIFS.

What can you do with the web interface in managing Proxmox Backup Server?

In the web interface, you can configure backups, settings, and restore data. In the interface you can configure network traffic settings, user access, and 2FA.

How does Proxmox Backup Server handle network load during backup operations?

It includes features for managing network load, such as bandwidth throttling. This ensures that backup operations do not overwhelm the network, maintaining optimal performance.

What are the best practices for configuring user permissions in Proxmox Backup Server?

Configuring user permissions involves assigning roles and access rights based on user responsibilities and requirements. This ensures that users have appropriate access to backup functions while maintaining data security.

How does Proxmox Backup Server perform quick data restoration?

With features like the file restore button and snapshot mode, it enables quick and efficient data restoration. These capabilities are crucial for minimizing downtime in case of data loss.

Proxmox Backup Server as a Comprehensive Solution for Backup and Restore

Proxmox Backup Server is a solution that provides Proxmox administrators what they need to protect critical data. It is also great for those with home lab environments to protect VMs and containers they don't want to lose or configurations that are hard to reproduce. As shown, the solution is not hard to install. It is easy to set up backup jobs, and you can quickly restore your backups in the Proxmox VE environment.

Proxmox SDN Configuration Step-by-Step

March 20, 2024
Proxmox

Proxmox sdn configuration

With the release of Proxmox 8.1, Proxmox introduced new networking features in the way of Proxmox SDN, or “software defined networking” that is fully integrated out of the box for use in the datacenter. Thanks to virtualization infrastructure, Software defined networking allows taking networking into software without having the need for physical network devices to spin up new networks, subnets, IP ranges, DHCP servers, etc. Proxmox SDN allows creating these virtualized network infrastructures. This post will look at Proxmox SDN configuration step-by-step and how it is setup.

Table of contents

Introduction to Proxmox SDN

Virtualization is not just for compute and storage or SD-WAN. Proxmox SDN is a new feature in Proxmox VE that allows you to create virtualized networks and isolated private network configurations in code. Think of it like creating your own little switch in software. These network are made up of virtual zones and networks (VNets) for communication. Using SDN, admins have much better control over networking management and virtual networks that are attached to VM guests and it is all free and open-source.

Note the following components of Proxmox software-defined network:

- **Zones** – a virtually separated network configuration or area
- **Virtual networks (VNets)** – Virtual network that is part of a zone
- **Subnets** – The network IP space insde a VNet.

Comparison with VMware NSX

You have probably heard about VMware's SDN solution called VMware NSX. There are many similarities with NSX and Proxmox SDN in capabilities. Arguably VMware NSX is a more robust solution that is a paid add-on to VMware vSphere. However, the Proxmox SDN solution is not as mature as VMware NSX that has been around for years now. I would like to see some of the additional micro-segmentation firewall features added to Proxmox SDN that we have in VMware NSX to create any number of connectivity rules and it can be integrated with ID sources for users, like AD domain configurations.

Use Cases of Proxmox SDN

What is the application of this technology? Using these components, you can create complex overlay networks on top of your existing network. The SDN network is a layer above the physical IP network where physical devices and hosts are connected.

Also, you can create your own isolated private network on each Proxmox VE server and span this to networks across multiple Proxmox VE clusters in many different locations.

Prerequisites

While Proxmox version 8.1 has the SDN components preloaded and the integration is available, according to the documentation, you will need to load the SDN package in Proxmox 7.X for every node in the cluster config:

```
apt update
apt install libpve-network-perl
```

After installation, you need to ensure that the following line is present at the end of the **/etc/network/interfaces configuration** file on all nodes:

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

Proxmox requires the dnsmasq package for SDN functionality to enable features like DHCP management and network addressing. To install the DNSmasq packages:

```
apt update
apt install dnsmasq
# disable default instance
systemctl disable --now dnsmasq
```

For advanced routing:

```
apt update
apt install frr-pythontools
```

Setting Up Proxmox SDN

Let's take a look at setting up software defined networking SDN on a Proxmox host and enabling an existing local Linux machine to connect. In this overview, we will enable automatic DHCP on the network interface so the machine can pull an IP from the IP range.

To Install Proxmox SDN as a simple network, we will do that in the following order:

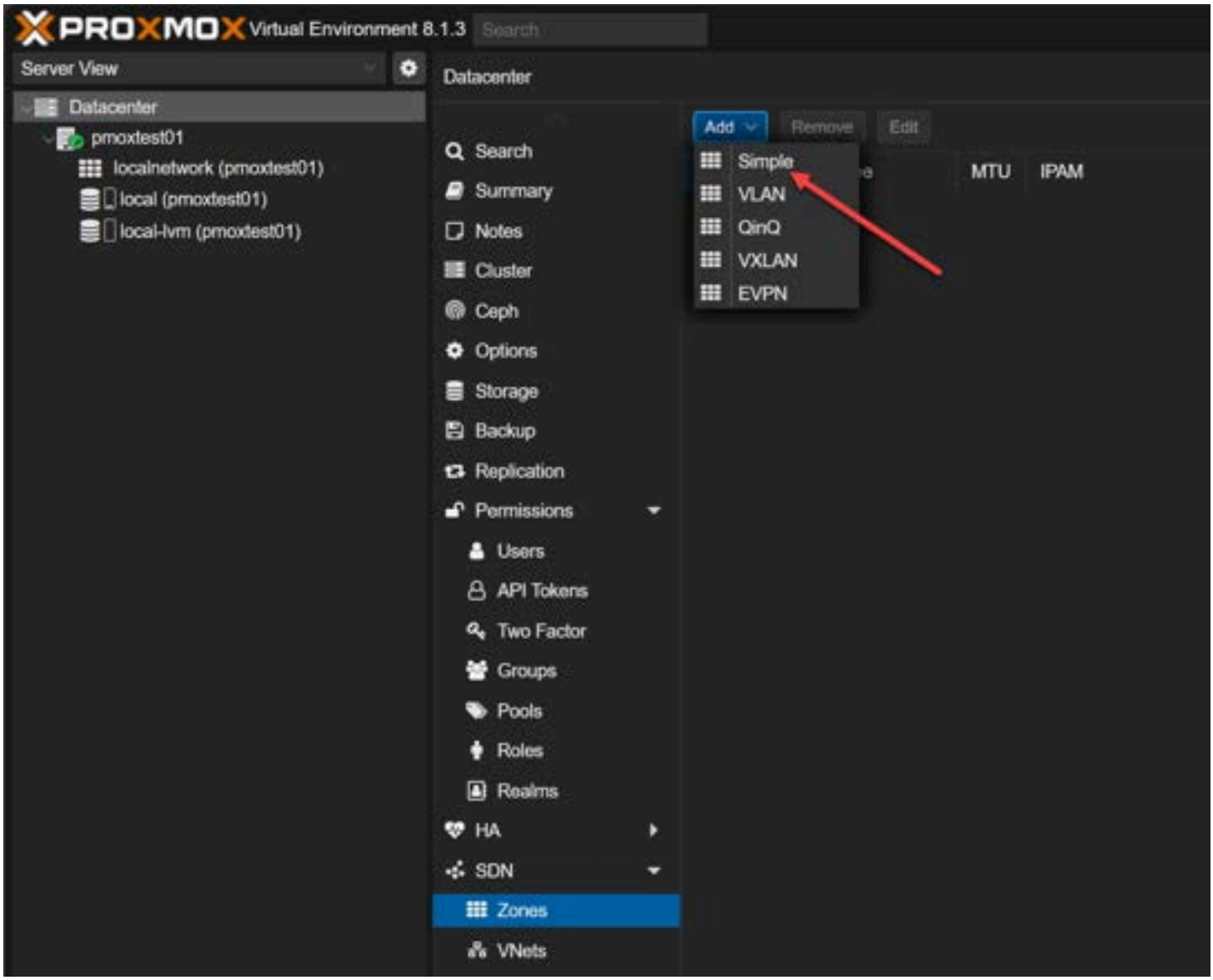
1. Create a **Simple** SDN Zone
2. Create a VNet
3. Create a Subnet and DHCP range
4. Apply the SDN configuration

1. Create a Simple SDN Zone

There are a few types of Zones you can create. These include:

- **Simple**: The simple configuration is an Isolated Bridge that provides a simple layer 3 routing bridge (NAT)
- **VLAN**: Virtual LANs enable the traditional method of dividing up a LAN. The VLAN zone uses an existing local Linux or OVS bridge to connect to the Proxmox VE host's NIC
- **QinQ**: Stacked VLAN (IEEE 802.1ad)
- **VXLAN**: Layer 2 VXLAN network that is created using a UDP tunnel
- **EVPN** (BGP EVPN): VXLAN that uses BGP to create Layer 3 routing. In this config, you create exit nodes to force traffic through a primary exit node instead of using load balancing between nodes.

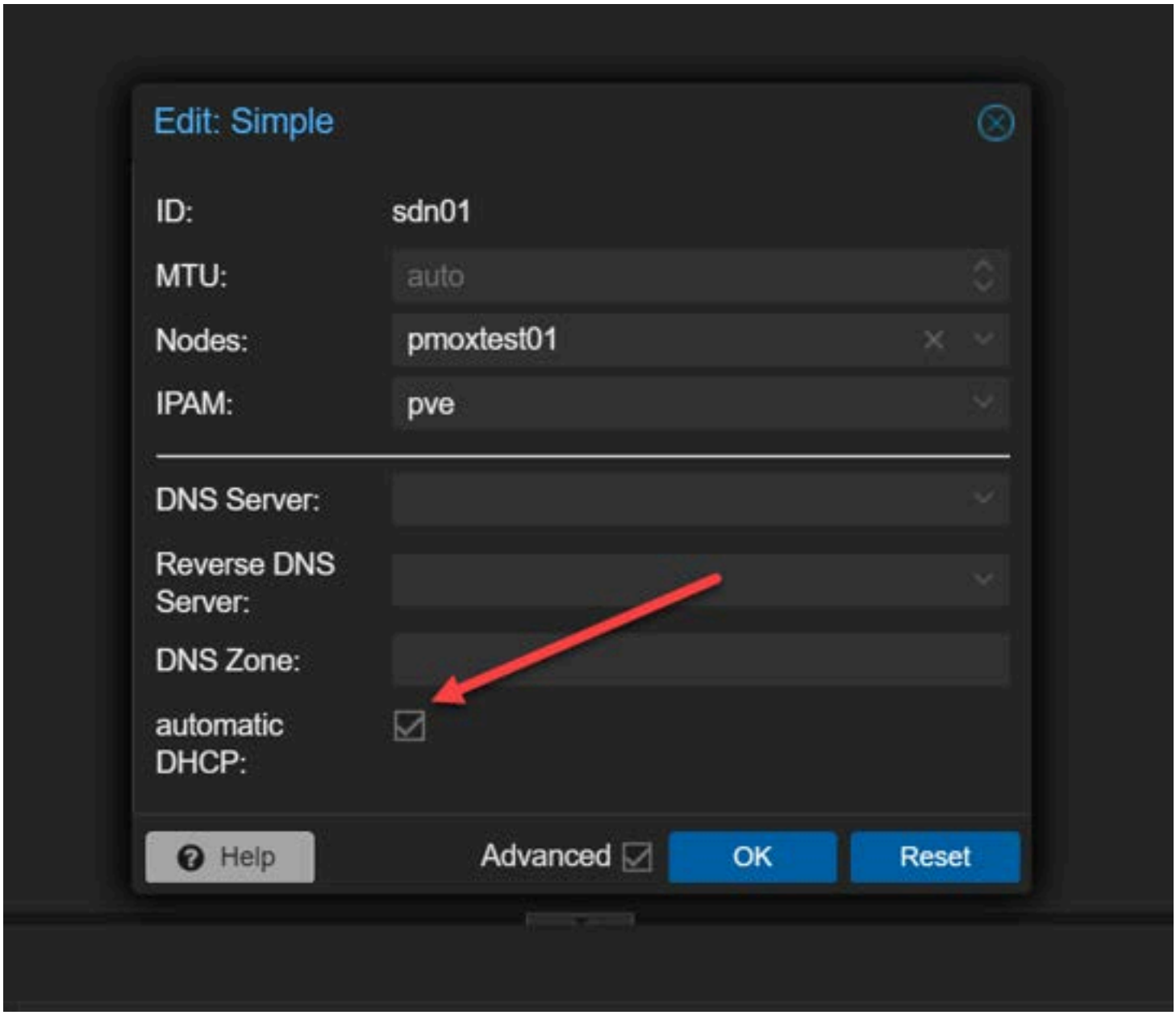
First, we need to create a new Zone. For this walkthrough, we will just be creating a **Simple** Zone. Login to your Proxmox node in a browser as root for the proper permissions. At the datacenter level, navigate to **SDN > Zones > Add.**



aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaCreating a new zone in proxmox sdn

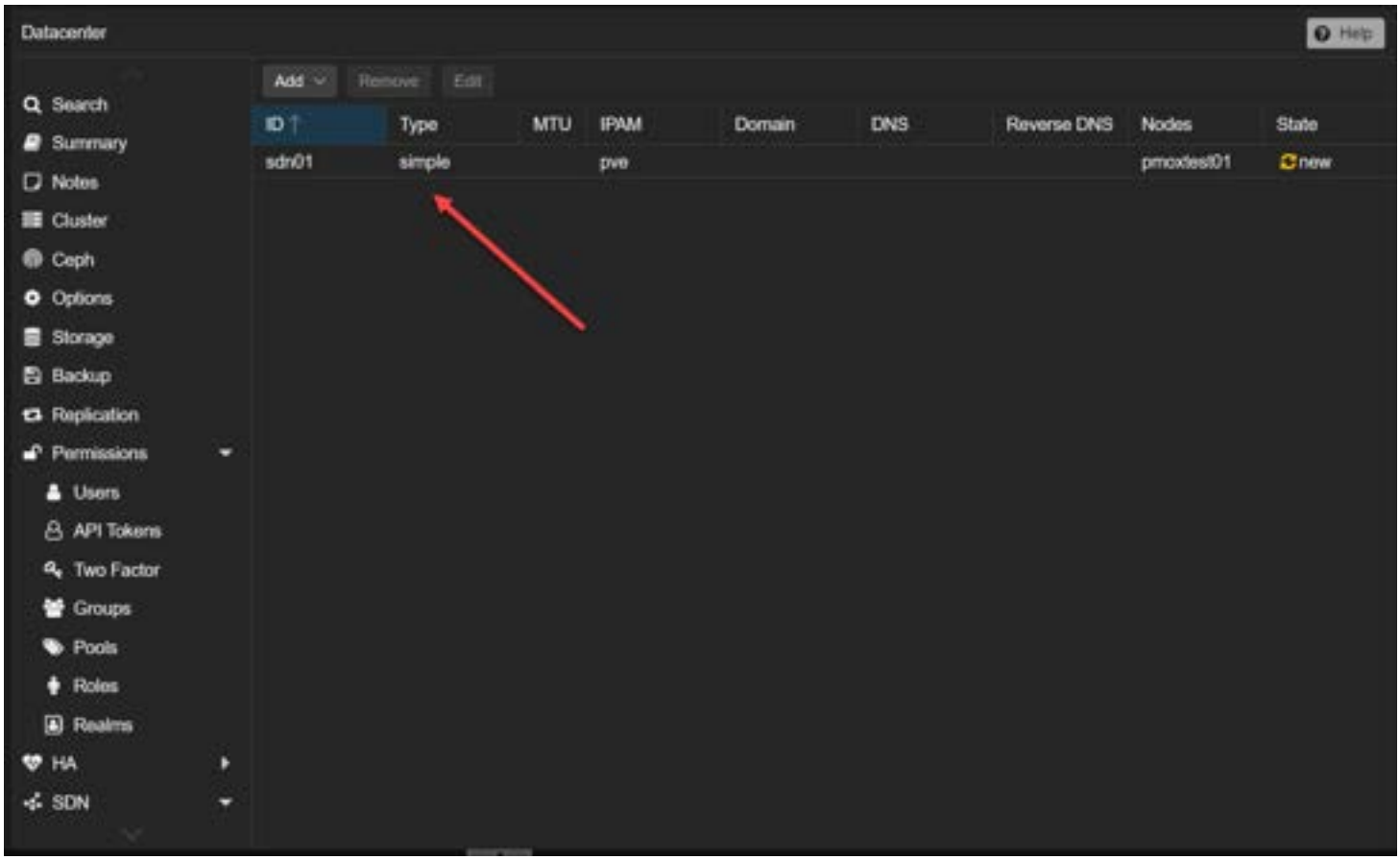
The SDN Zone configuration also allows you to set the zone for **automatic DHCP** configuration that will allow your VMs to pull an IP address from the VNet and Subnet configuration we will setup below. You can also set the MTU value for the size of the ethernet frames (packet), and DNS configuration, including DNS server, DNS zone, etc. In this example, I am creating a SDN Zone called **sdn01**.

The MTU value is important to note as with VXLAN, it uses 50 bytes to encapsulate the packet, you need to reduce the size by 50 bytes less than the normal MTU value. Optional will default to a size of 1450 on auto. In the case of VXLAN with IPSEC security, customers will need to reduce the MTU by 60 with IPv4, or 60 for IPv6 for guest traffic or you will see an issue with connectivity that may be a problem that is hard to uncover.



aaaaaaEnabling automatic dhcp

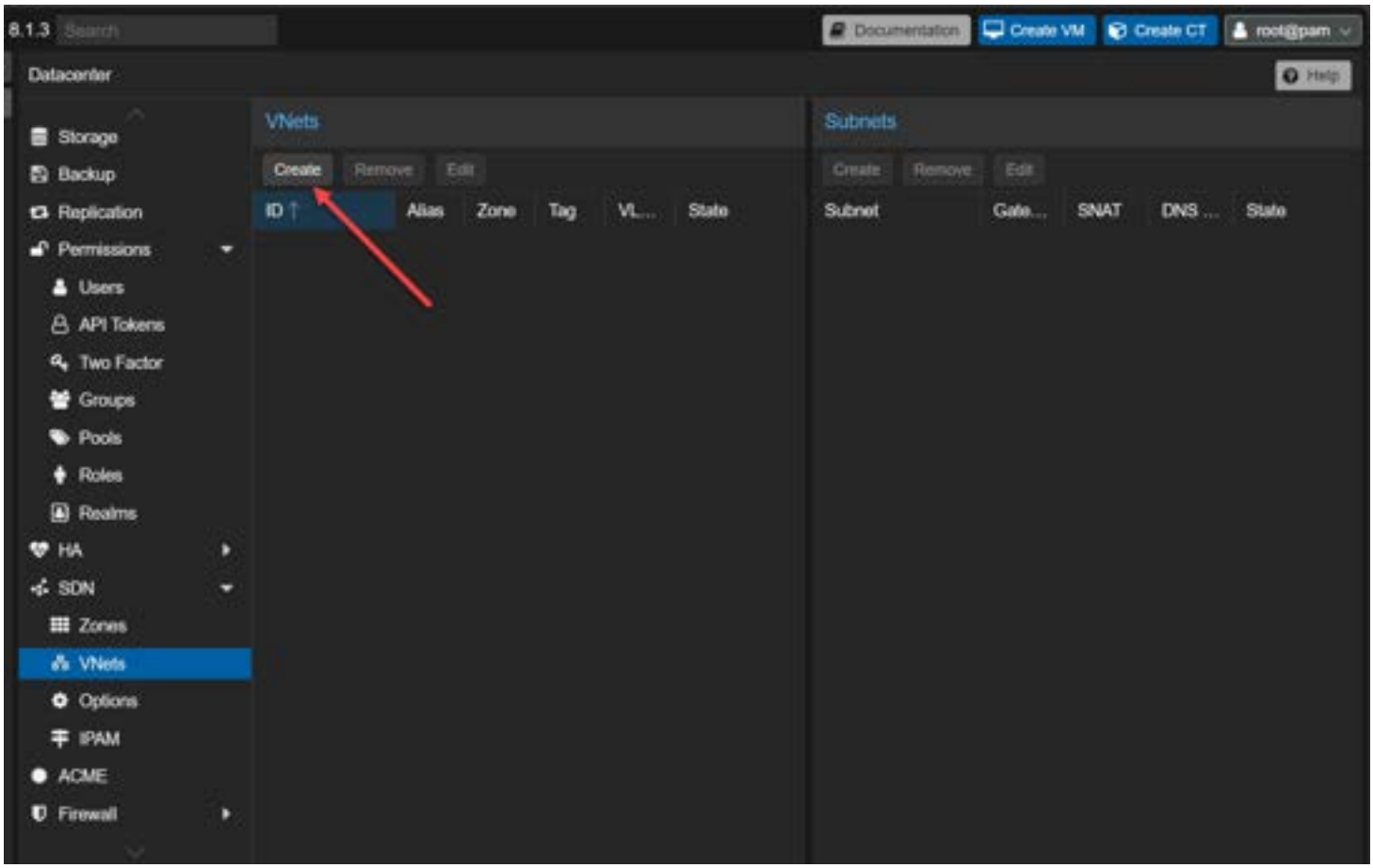
After clicking OK above, we see the new **sdn01** Simple Zone.



Viewing the simple zone in proxmox sdn

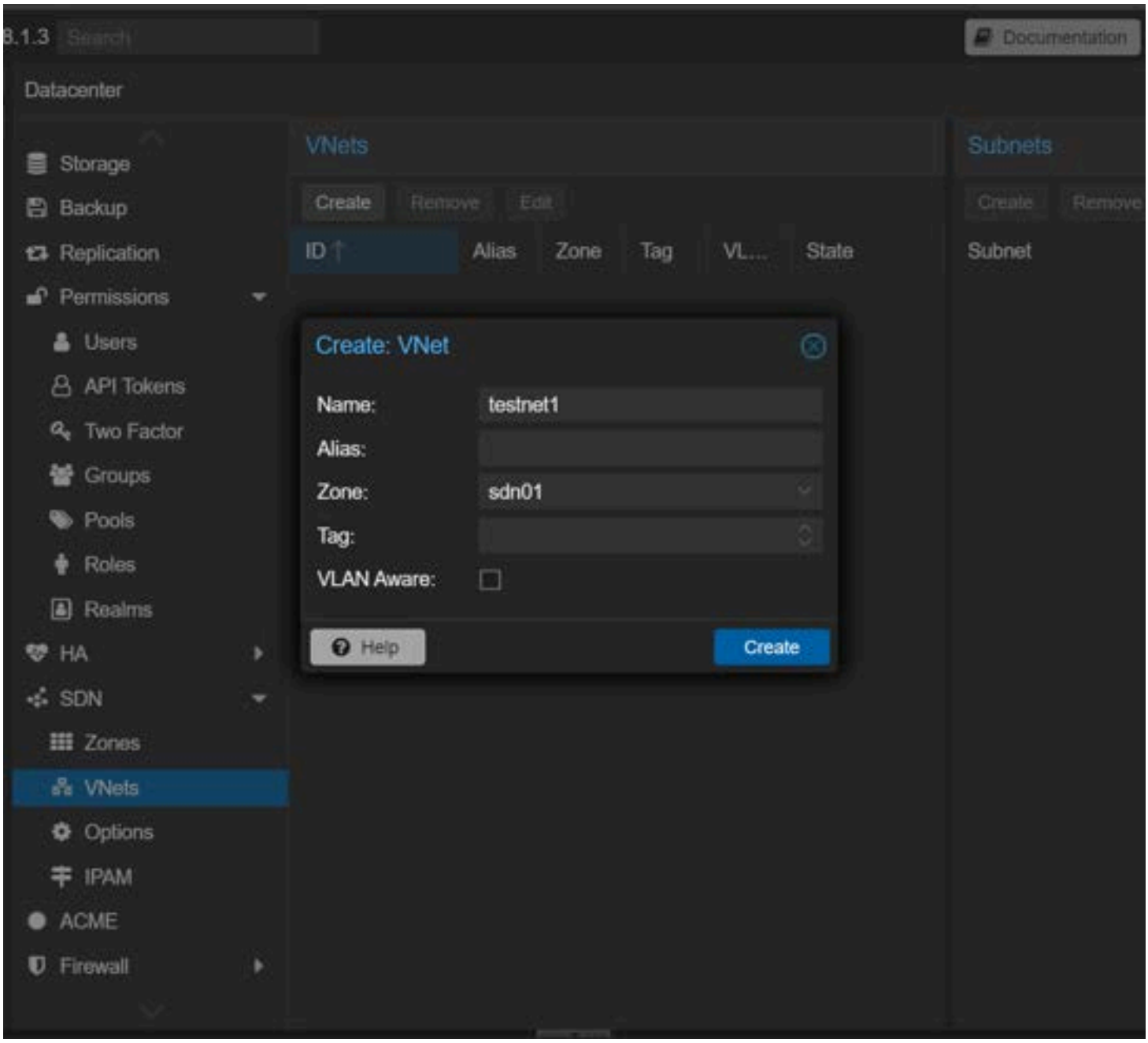
2. Create a VNet

Next, we need to create a VNet in PVE. Navigate to the VNet menu under the SDN menu and click to **Create** a new VNet.



Beginning the process to create a new vnet

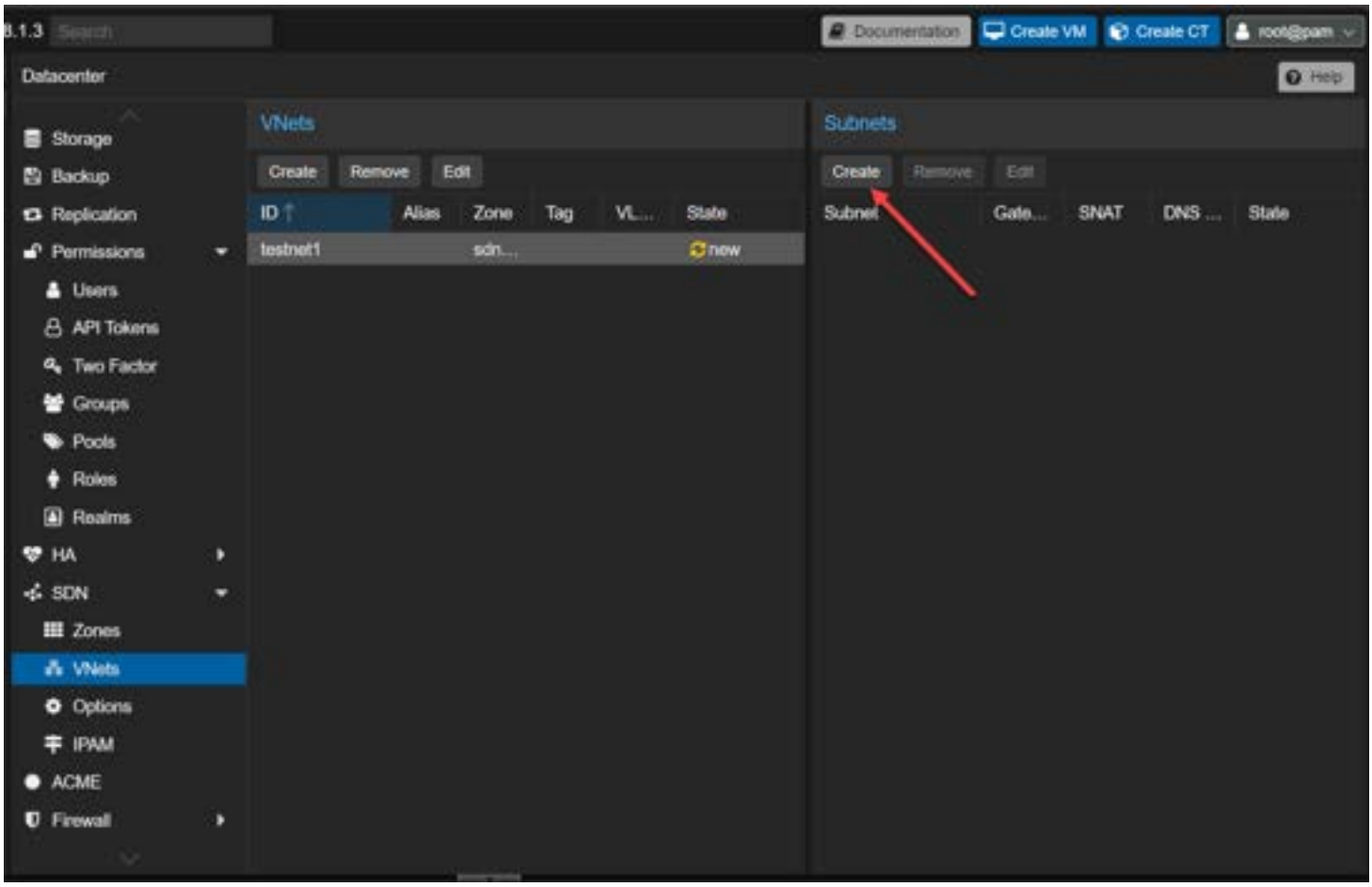
Create a name for the VNet and select the Zone we created above. You also have the option to make these VLAN aware with a tag and also create an alias.



Configuring the new vnet in proxmox sdn

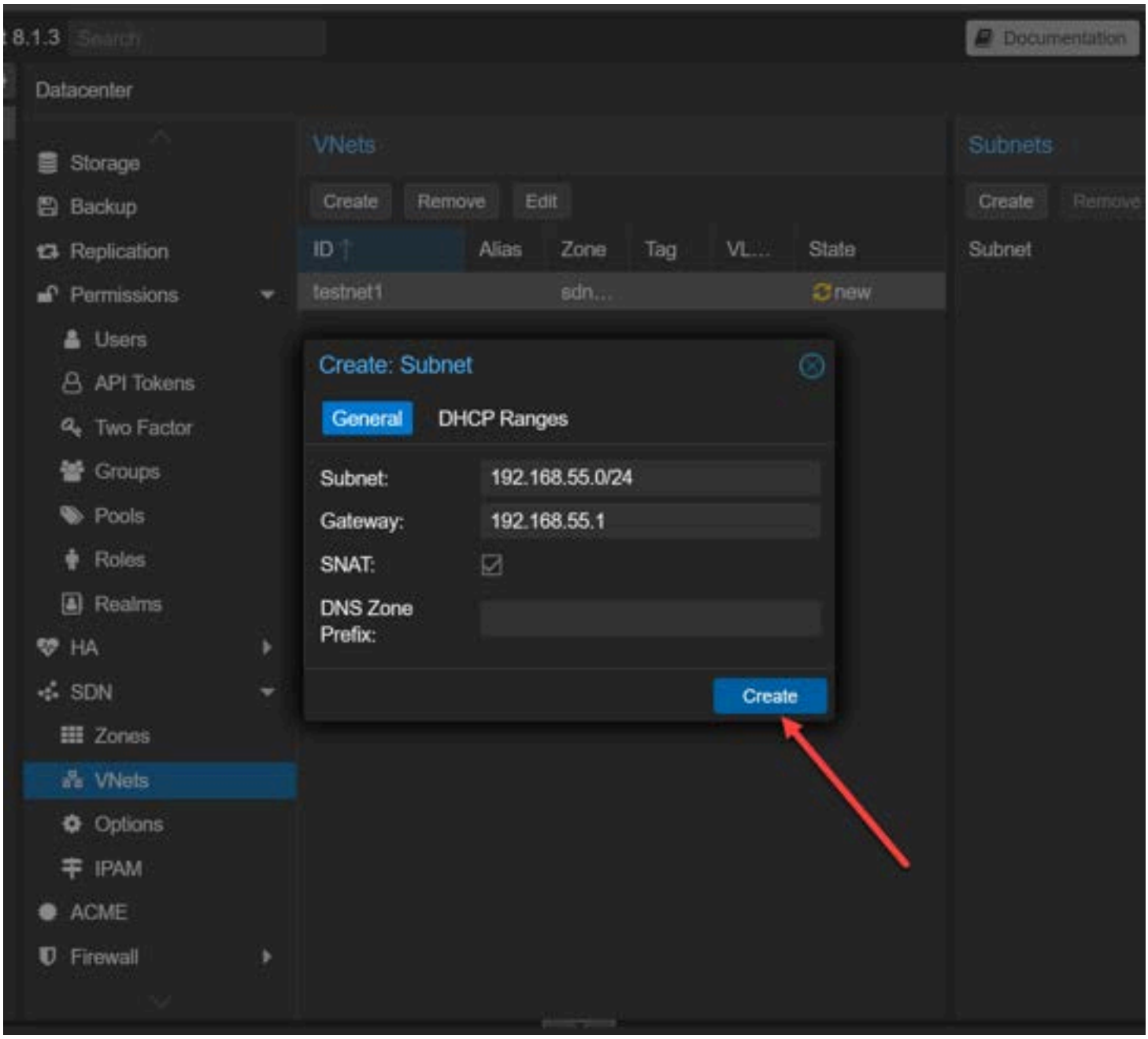
3. Create a Subnet and DHCP range

After creating the VNet, we can create a Subnet. Click the **Create** button on the **Subnets** screen.



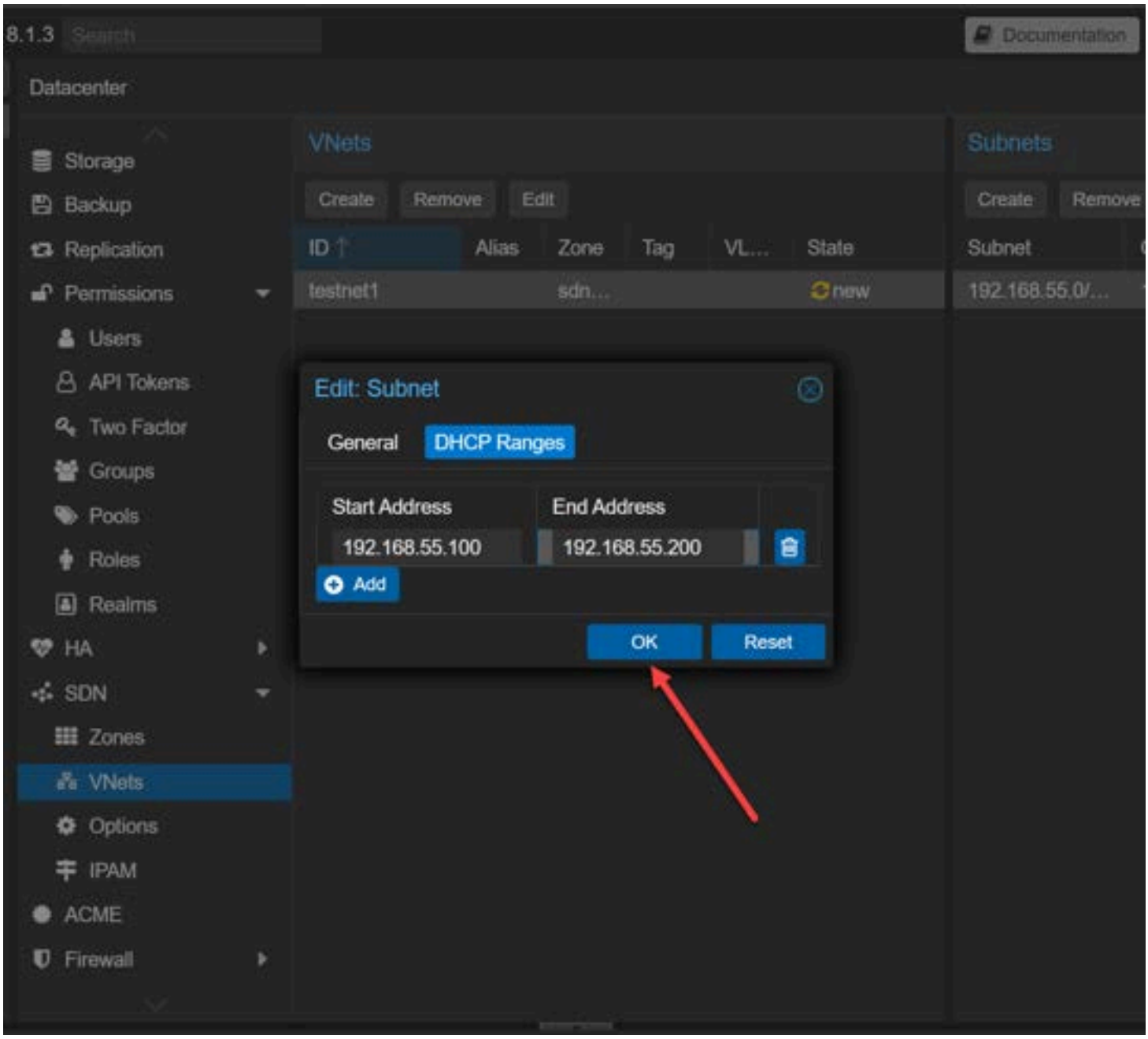
Creating a new subnet in proxmox sdn

Enter your IP address CIDR information and Gateway. If you populate the Gateway here, your Proxmox server will assume this IP address. Also, you can check the **SNAT** box. This will allow your VMs connected to the SDN network to easily connect to external networks beyond the SDN network (aka the Internet and your physical network) by masquerading as the IP and MAC of the host. Click **Create**.



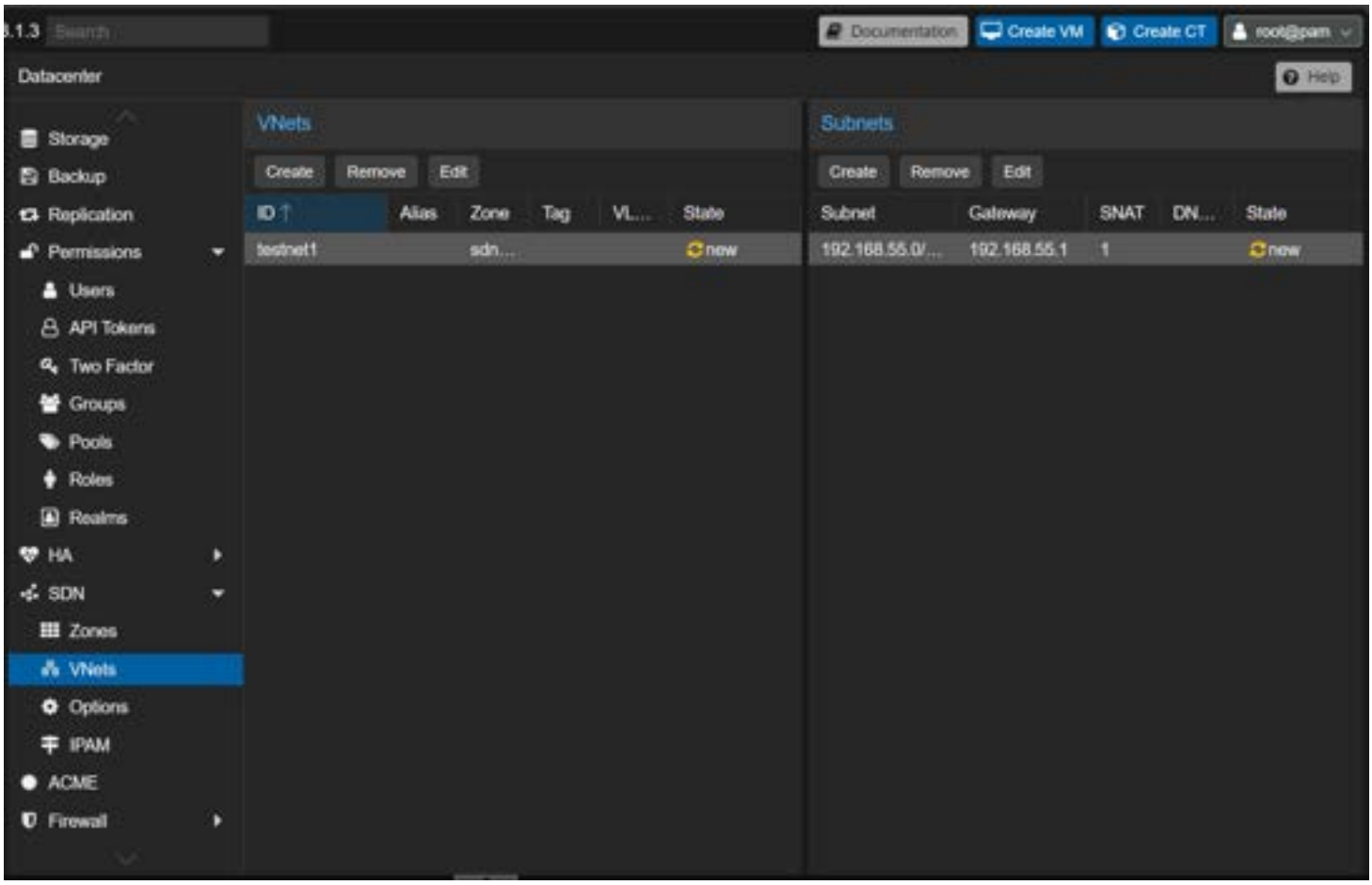
Creating a new subnet

Click on the **DHCP Ranges** and enter your start and end address for the DHCP range. It will hand out addresses from this range of IPv4 IPs.



Creating a dhcp range in proxmox sdn

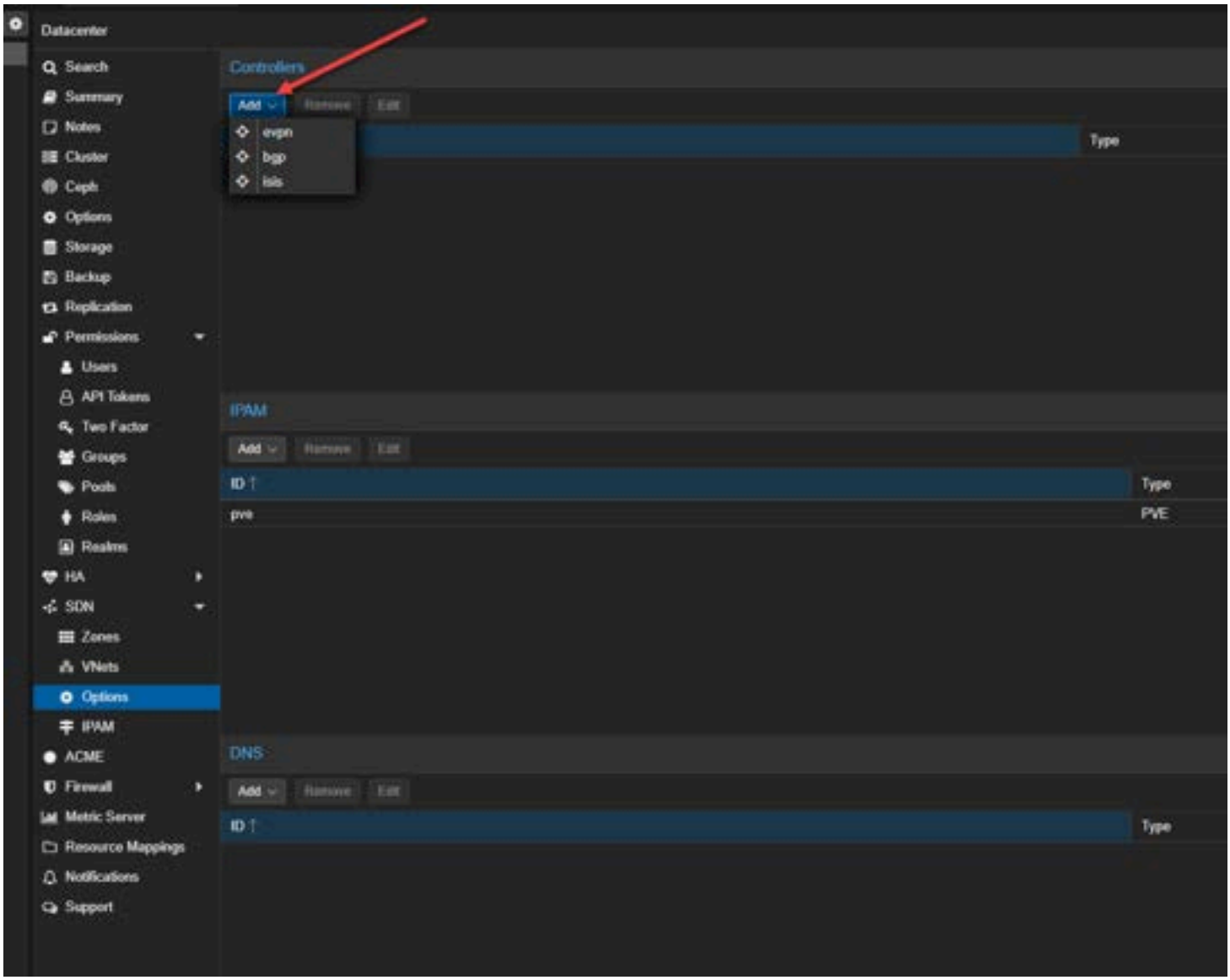
After clicking OK, we will see the new VNet and Subnet displayed.



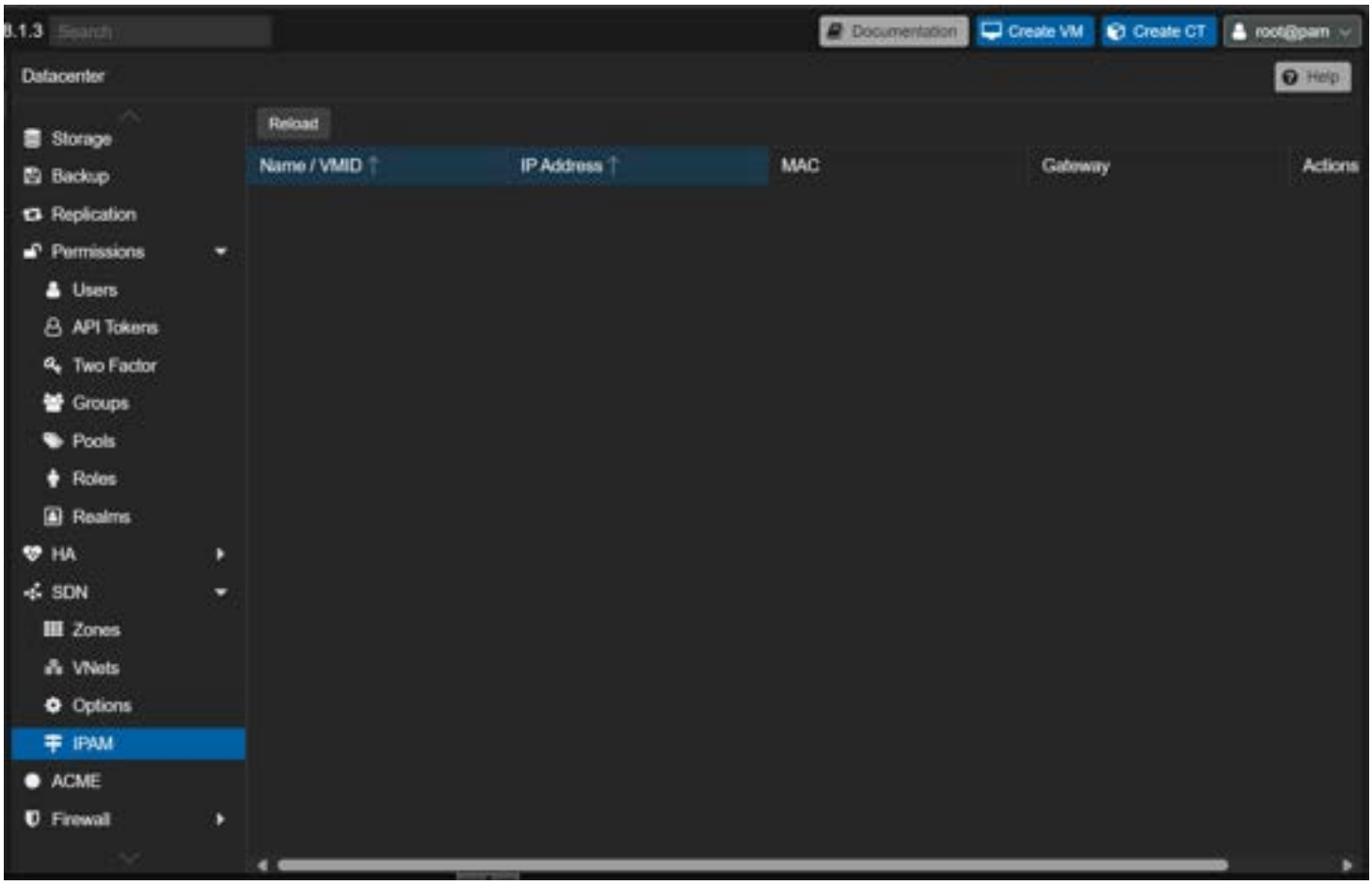
Looking at the vnets and subnets created

We are not setting anything in the **Options** screen or **IPAM**. However, let's take a look at what those screens look like. Under the Options screen and the **Controllers** section, we can add network controllers for more advanced configurations like VXLAN to configure network tunnel configurations between peers, which are the Proxmox nodes. Under the Controllers section, we can add **EVPN**, **EBGP**, and **ISIS.**

For BGP controllers, these are not used directly by a zone. You can configure FRR to manage BGP peers. BGP-EVPN configuration define a different ASN by node. When you click the controller dropdown, you will see a list of options.



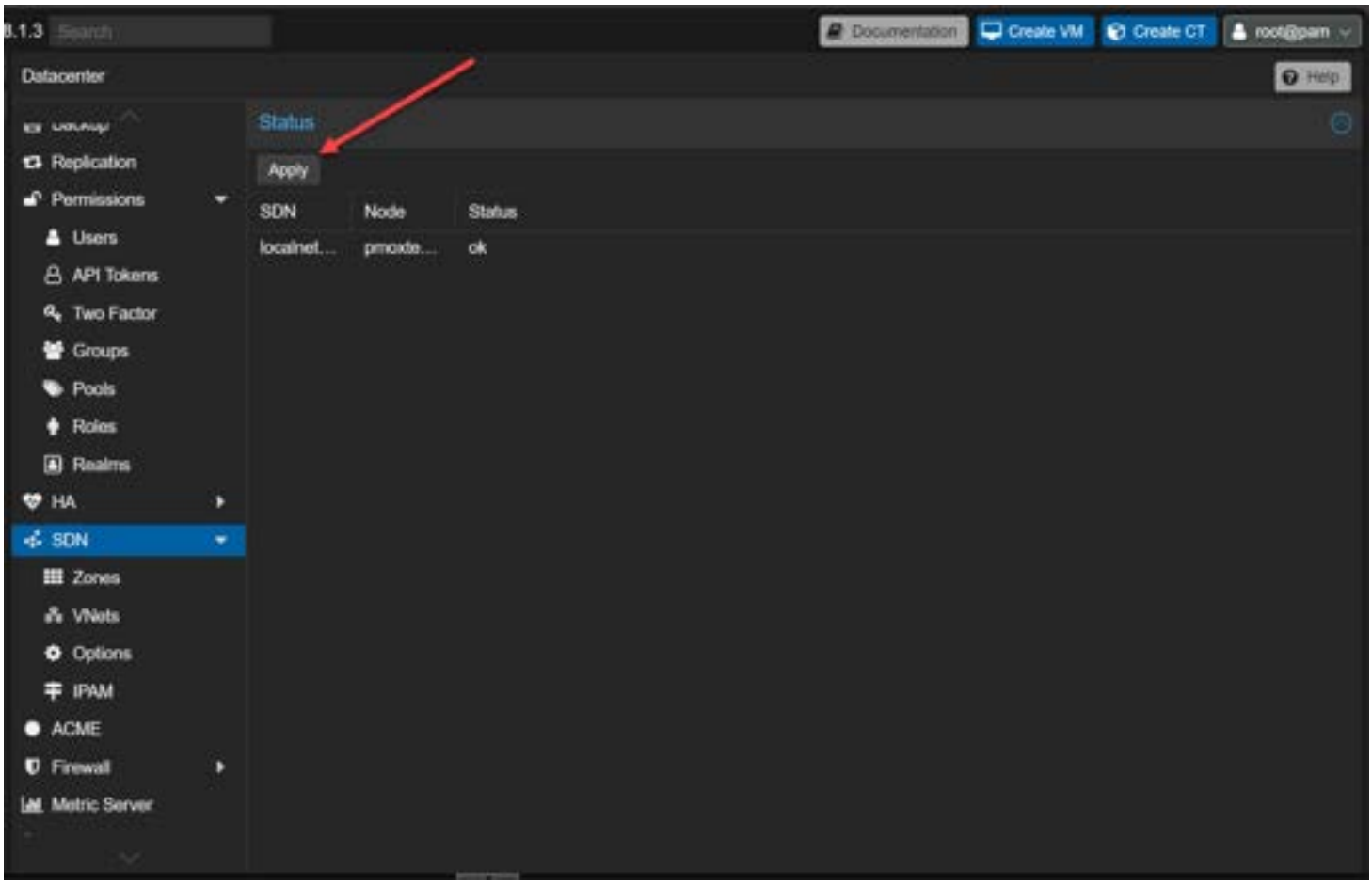
Looking at controllers and options available in proxmox sdn



Looking at ipam

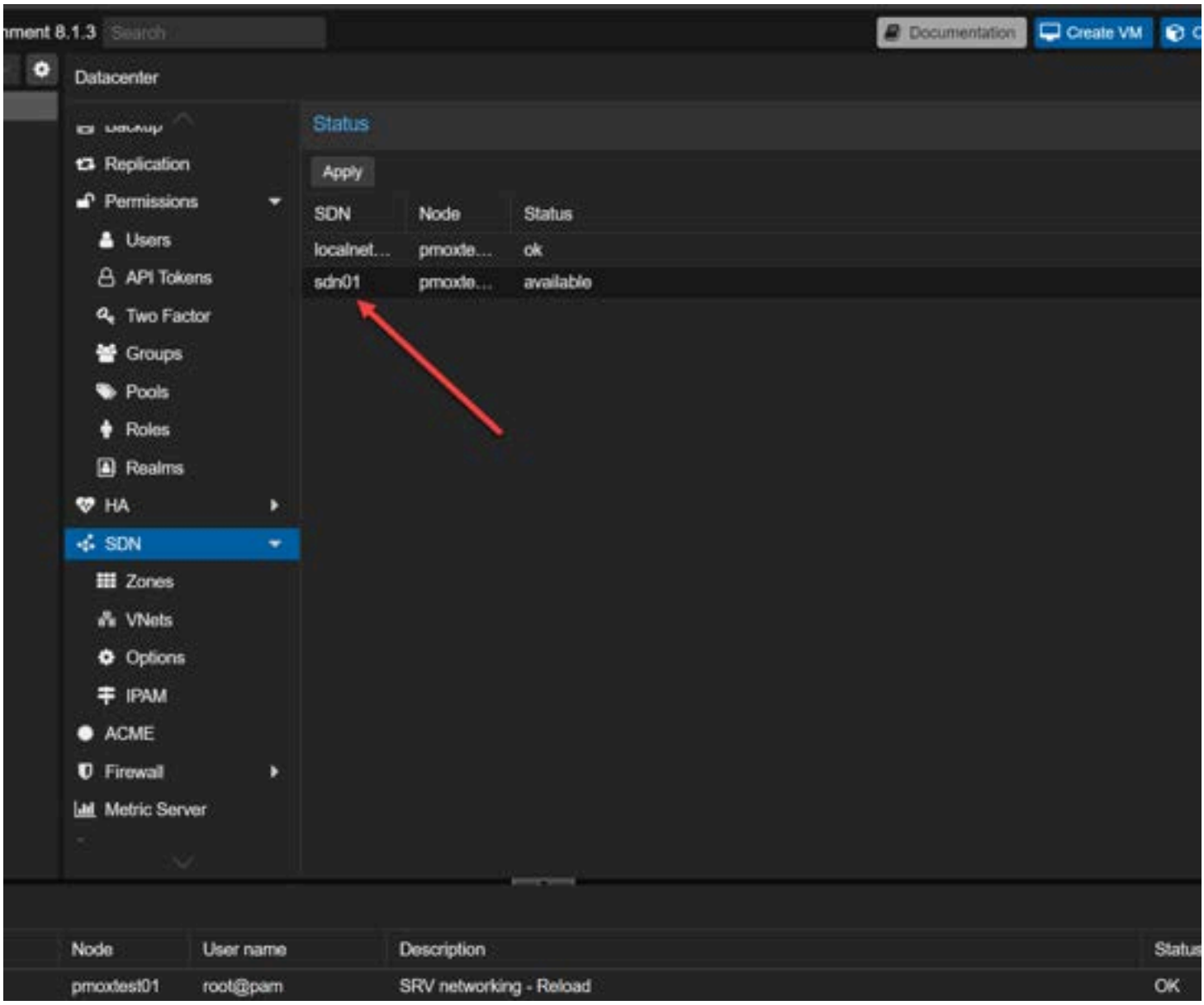
4. Apply the SDN configuration

It is very important to understand that creating the configuration we have created **does not apply** the configuration. It only **stages** the configuration so to speak. You need to click the **SDN** parent menu and click the **Apply** button.



Apply the proxmox sdn configuration

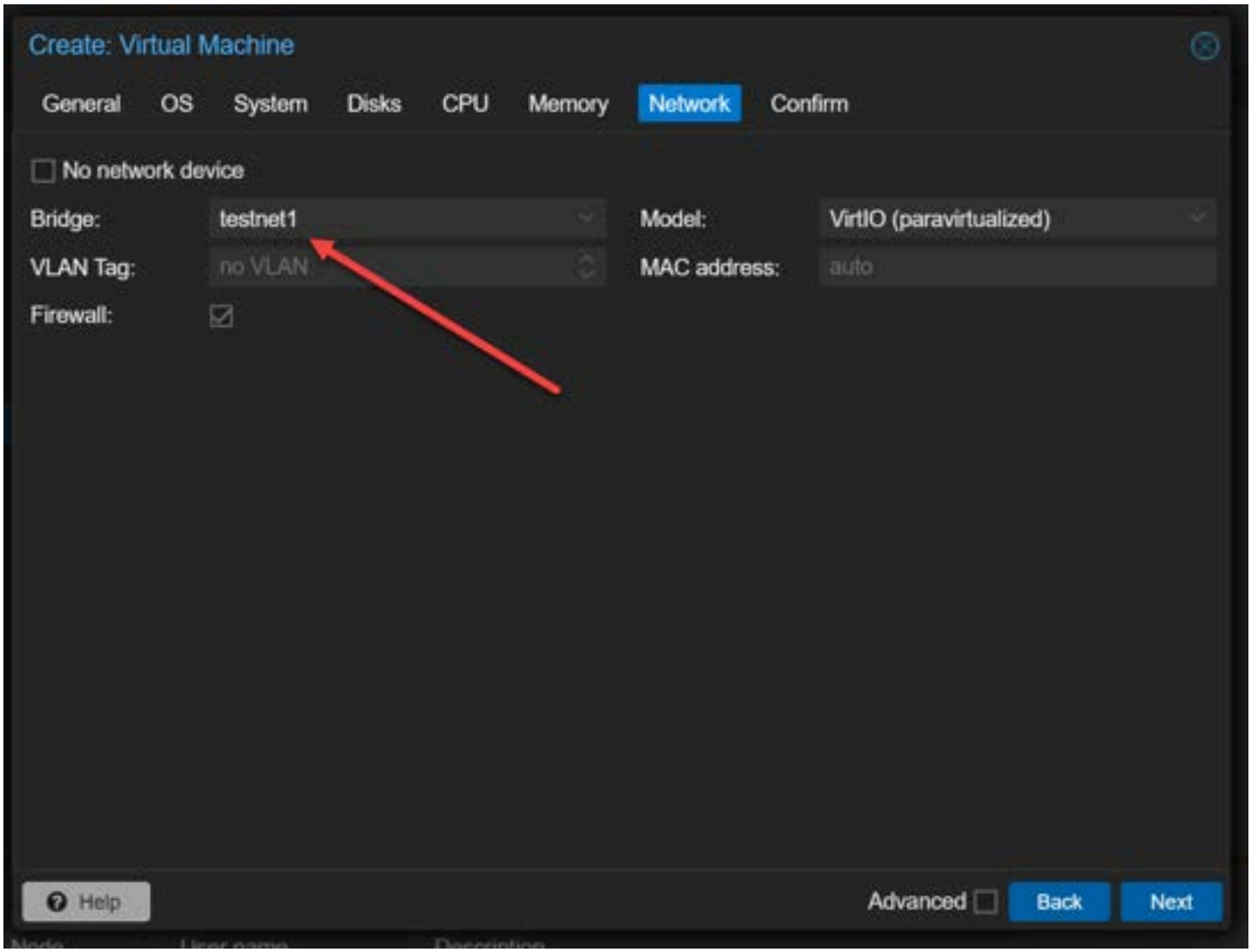
Now we see the new SDN network status after the configuration is applied and the Proxmox networking services are restarted.



Viewing the new configuration applied in proxmox

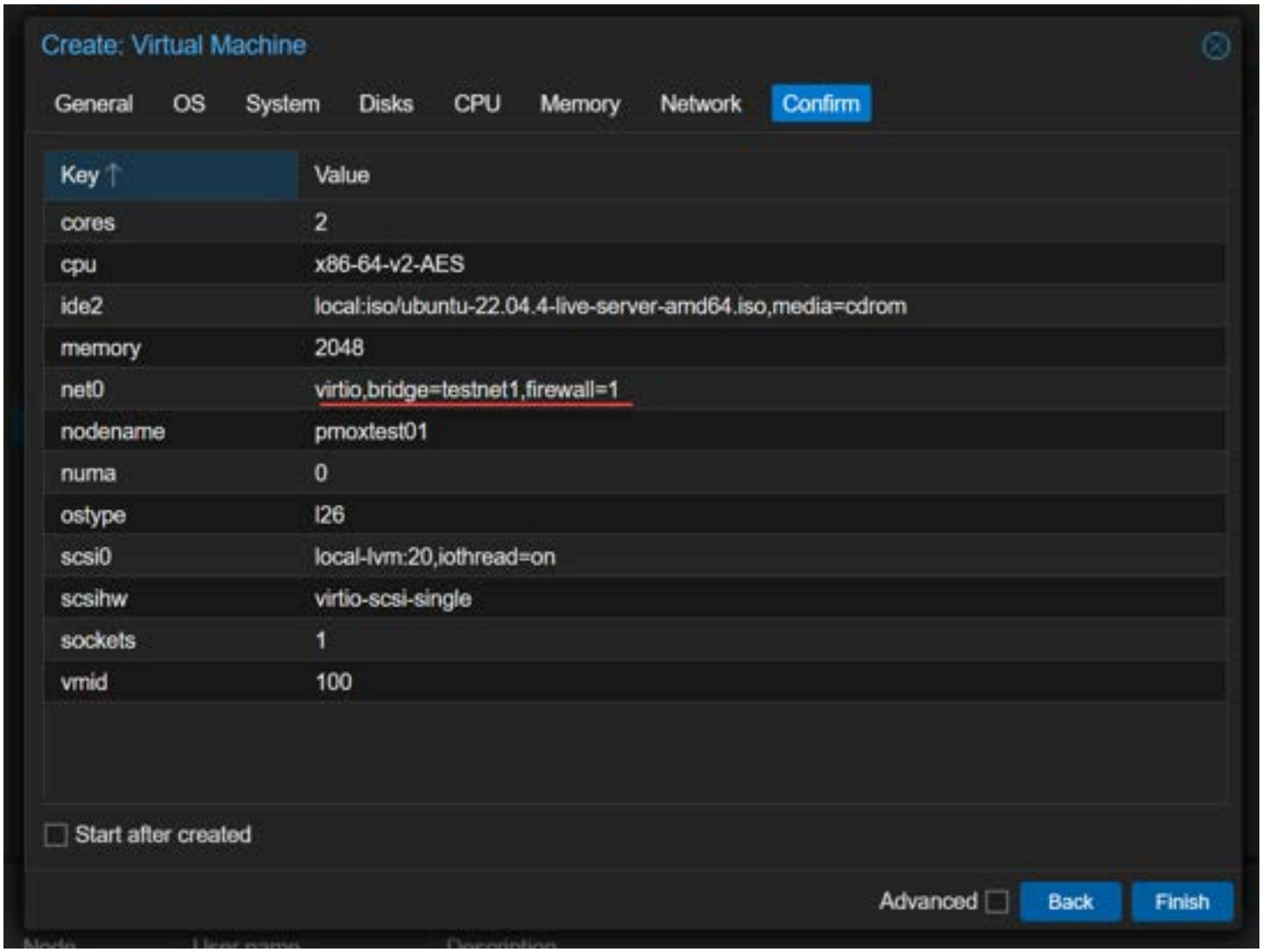
Connect Virtual Machines and Containers to the SDN network

Now that we have the configuration for SDN in place on our virtual switches bridge in the hypervisor, we can connect the virtual machine or container (CT) to the new SDN network.



Connecting a new virtual machine to the proxmox sdn network

Below, you see the summary screen of creating a new virtual machine and we see I have connected it to the new SDN network.



Summary of new vm creation details

After installing Ubuntu, the VM correctly grabs a DHCP address from the range configured. Also, we can ping the gateway that was established in the configuration. Keep in mind how cool this really is. We have a network with total separation from the other physical network technologies for VM traffic and it is totally defined in software.

```
linuxadmin@ubuntu01:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens18: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether bc:24:11:01:d6:ab brd ff:ff:ff:ff:ff:ff
    altname enp0s18
    inet 192.168.55.100/24 metric 100 brd 192.168.55.255 scope global ens18
       valid_lft forever preferred_lft forever
    inet6 fe80::be24:11ff:fe01:d6ab/64 scope link
       valid_lft forever preferred_lft forever
linuxadmin@ubuntu01:~$ ping 192.168.55.1
PING 192.168.55.1 (192.168.55.1) 56(84) bytes of data.
64 bytes from 192.168.55.1: icmp_seq=1 ttl=64 time=0.709 ms
64 bytes from 192.168.55.1: icmp_seq=2 ttl=64 time=0.359 ms
64 bytes from 192.168.55.1: icmp_seq=3 ttl=64 time=0.390 ms
64 bytes from 192.168.55.1: icmp_seq=4 ttl=64 time=0.438 ms
64 bytes from 192.168.55.1: icmp_seq=5 ttl=64 time=0.383 ms
64 bytes from 192.168.55.1: icmp_seq=6 ttl=64 time=0.395 ms
64 bytes from 192.168.55.1: icmp_seq=7 ttl=64 time=0.497 ms
64 bytes from 192.168.55.1: icmp_seq=8 ttl=64 time=0.348 ms
```

New virtual machine pulls a dhcp address from proxmox sdn

Key points to remember

Let's consider a few key points to remember about the Proxmox SDN solution.

Network Interfaces and VLAN Configuration

Network interfaces are the gateways between your virtual machines and the broader network (Internet). Make sure to give attention to detail to configure these correctly for proper connectivity and optimal performance.

VLANs enable you to segment your network into isolated sections. With VLANs you can create a secure, organized network zones.

VXLAN Zone Implementation

VXLAN zones extend VLAN capabilities and create overlay networks across even different physical network locations. With VXLAN, you can build a complex, scalable network architecture.

Advanced Proxmox SDN Features

Some of the advanced Proxmox SDN features include automatic DHCP assignment to IP address management. Understand how you can use these features to enhance your network management.

Virtual Zones and Traffic Isolation

Creating virtual zones within Proxmox SDN allows network traffic segregation. This enhances the security and performance of your network. Traffic isolation is crucial for security.

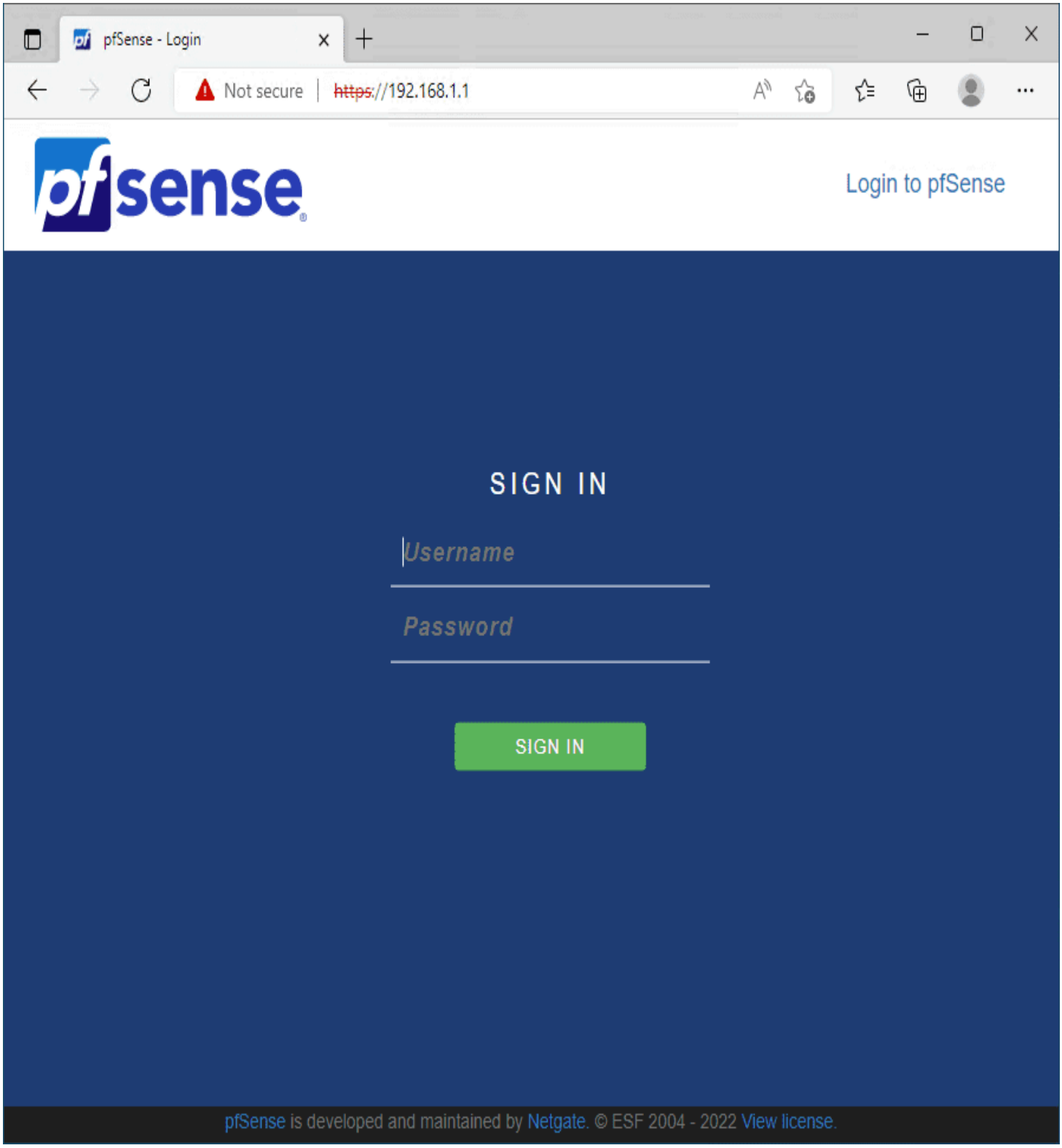
Wrapping up Proxmox SDN configuration

The new Proxmox SDN features in Proxmox 8.1 and above are a great new feature that allows you to create new networks quickly and easily in software. Networking has traditionally been a challenge to configure quickly and easily since physical network devices and configurations have to be changed. With SDN, all of this changes with the network overlay created. The underlying physical network no longer has to be updated, like network switches, or changed for new networks and connectivity to be created.

Proxmox SDN is easy to configure and you can create a simple new network as shown in the walkthrough to start playing around with the new feature in your home lab. Let me know in the comments or VHT forum if you have played around with Proxmox SDN as of yet and what use cases you are finding in the home lab.

pfSense Proxmox Install Process and Configuration

August 26, 2022
Proxmox



Logging into pfSense VM for the first time

Many great open source solutions are available these days for many use cases, including security, networking, routing, etc. Two of those include pfSense and Proxmox server. Proxmox VE is an open-source solution that you can easily download for free and run a pfSense VM for routing, virtual network interfaces, firewall capabilities, etc. Let's deep dive into the process of pfSense Proxmox install process and configuration and see what steps are involved.

pfSense on Proxmox installation and configuration - Step-by-step
https://youtube.com/watch?v=mwDv790YoZ0

What is Proxmox VE?

Proxmox VE is a open-source virtualization solution allowing you to run virtual machines, including pfSense VM solutions. This is great as it allows you to run a pfSense virtual machine that can perform routing, firewalling, VPN, and all the great features that pfSense includes as part of the solution. In addition, you can run other virtual machines along with pfSense in Proxmox.

You can also run a Proxmox cluster for the highest availability requirements and for failover purposes.

Running pfSense on Proxmox VE

Running pfSense on Proxmox server, pfSense Proxmox, is a great way to have powerful features for no cost, running on commodity bare metal hardware. Proxmox provides many enterprise hypervisor features, including backups that can be enabled for newly created virtual machine boxes running in Proxmox server.

Run on bare metal or virtual machine

Proxmox hosts can run on a bare metal server or run as a virtual machine itself. If you would like to see how to run Proxmox Server as a nested VMware virtual machine, check out my post here: [Nested Proxmox VMware installation in ESXi – Virtualization Howto]

What is pfSense?

First of all, what is pfSense? The pfSense solution is a secure and widely used firewall distribution that is available as a virtual machine appliance or running on hardware platforms from Netgate.

Either way, you can get network interfaces either in hardware or virtual machine network interfaces, allowing you to route, firewall, and connect traffic to your network as you would any other enterprise firewall solution.

Netgate hardware

Netgate hardware firewall appliances

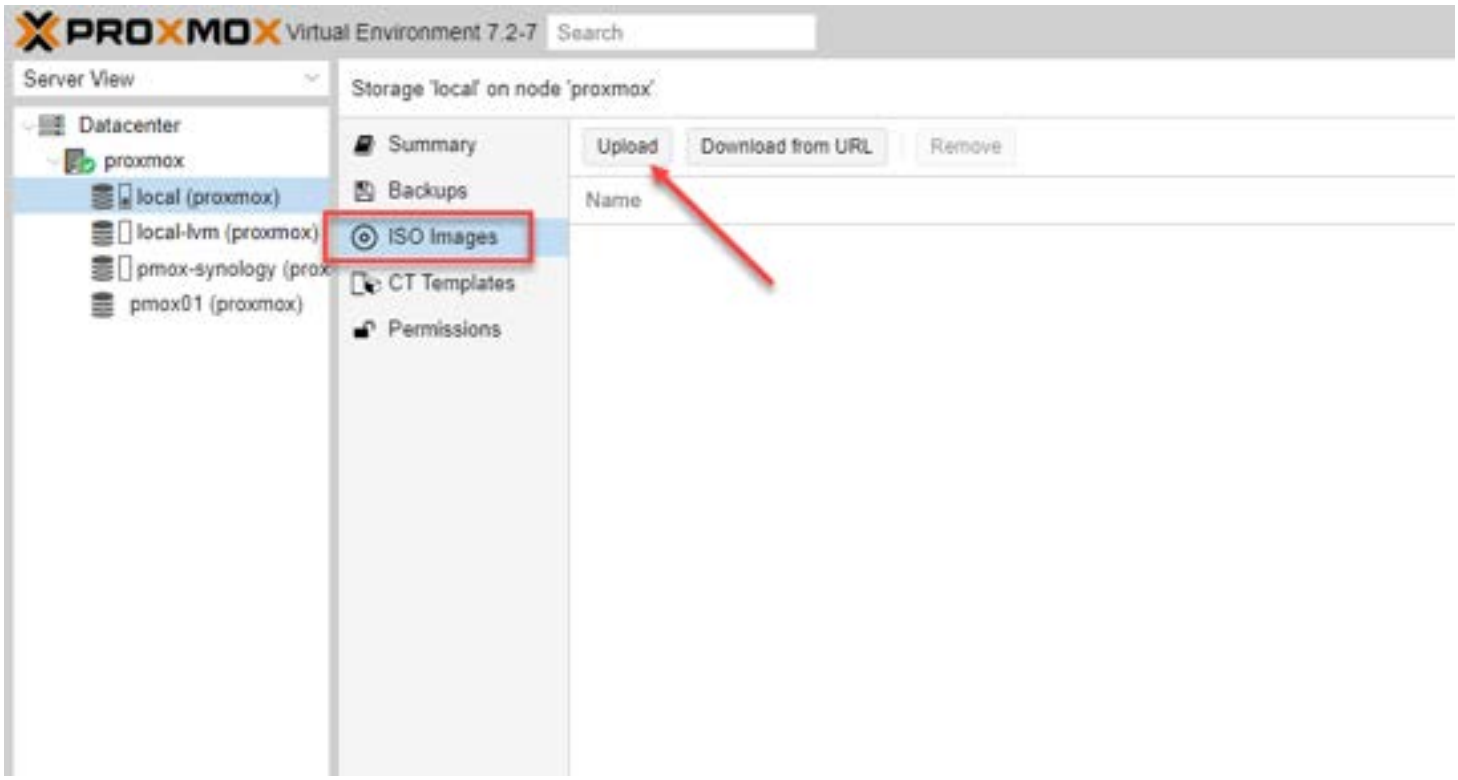
Download pfSense ISO image

You can download the pfSense ISO image here for the Community edition:

[Download pfSense Community Edition]

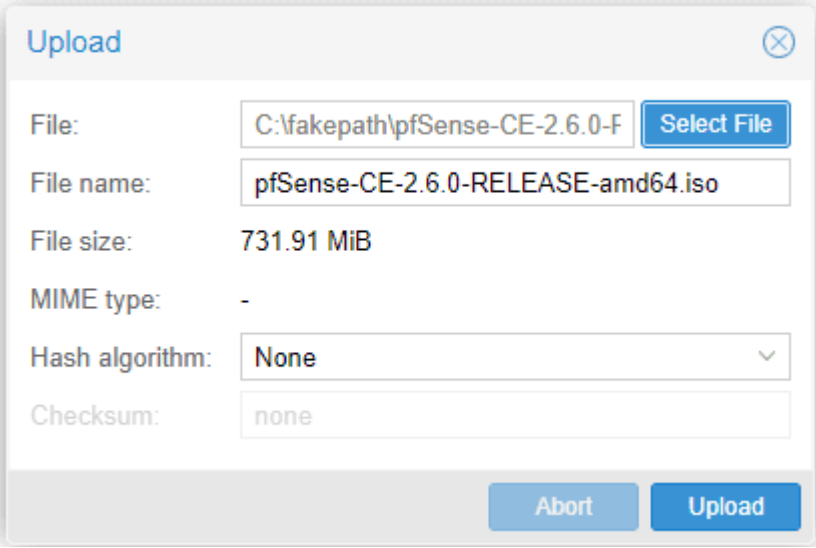
Upload ISO image to Proxmox server

Before we can run our pfSense VM installation on Proxmox ve, we need to get the [installation ISO image for pfSense VM uploaded to Proxmox] VE. To do that, we log into Proxmox VE and browse to our local Proxmox storage, select ISO images and click the **Upload** button



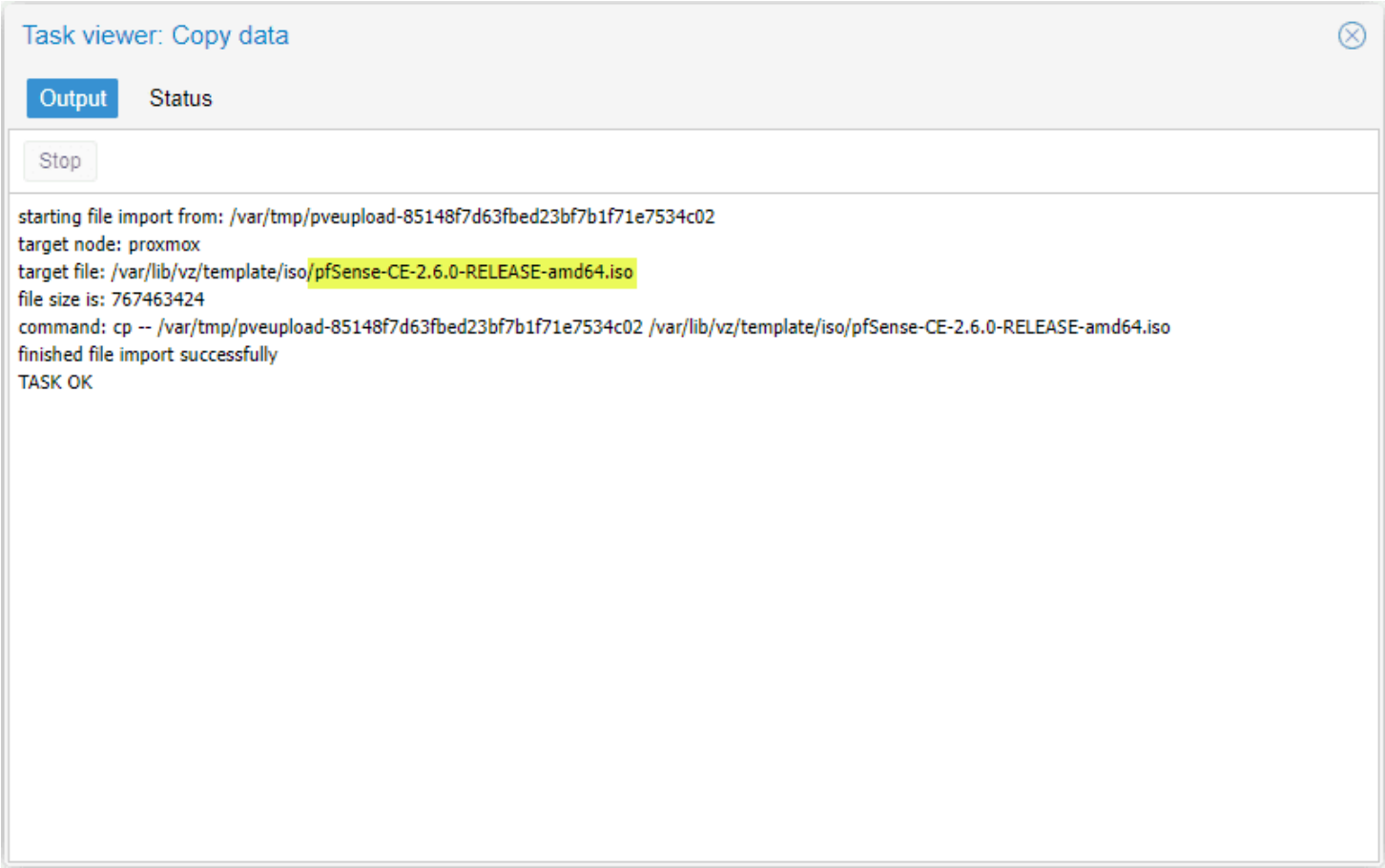
Beginning process to upload pfSense ISO to Proxmox

Once you click the Upload button, you will have the ability to click the **Select File** button. Click the Select File button and browse to your downloaded pfSense ISO image. Then, click **Upload**.



Choose the pfSense ISO for upload to Proxmox VE

After you click Upload, you will see the upload progress. Then, the screen below should display, noting the upload of the ISO image was successful for pfSense.

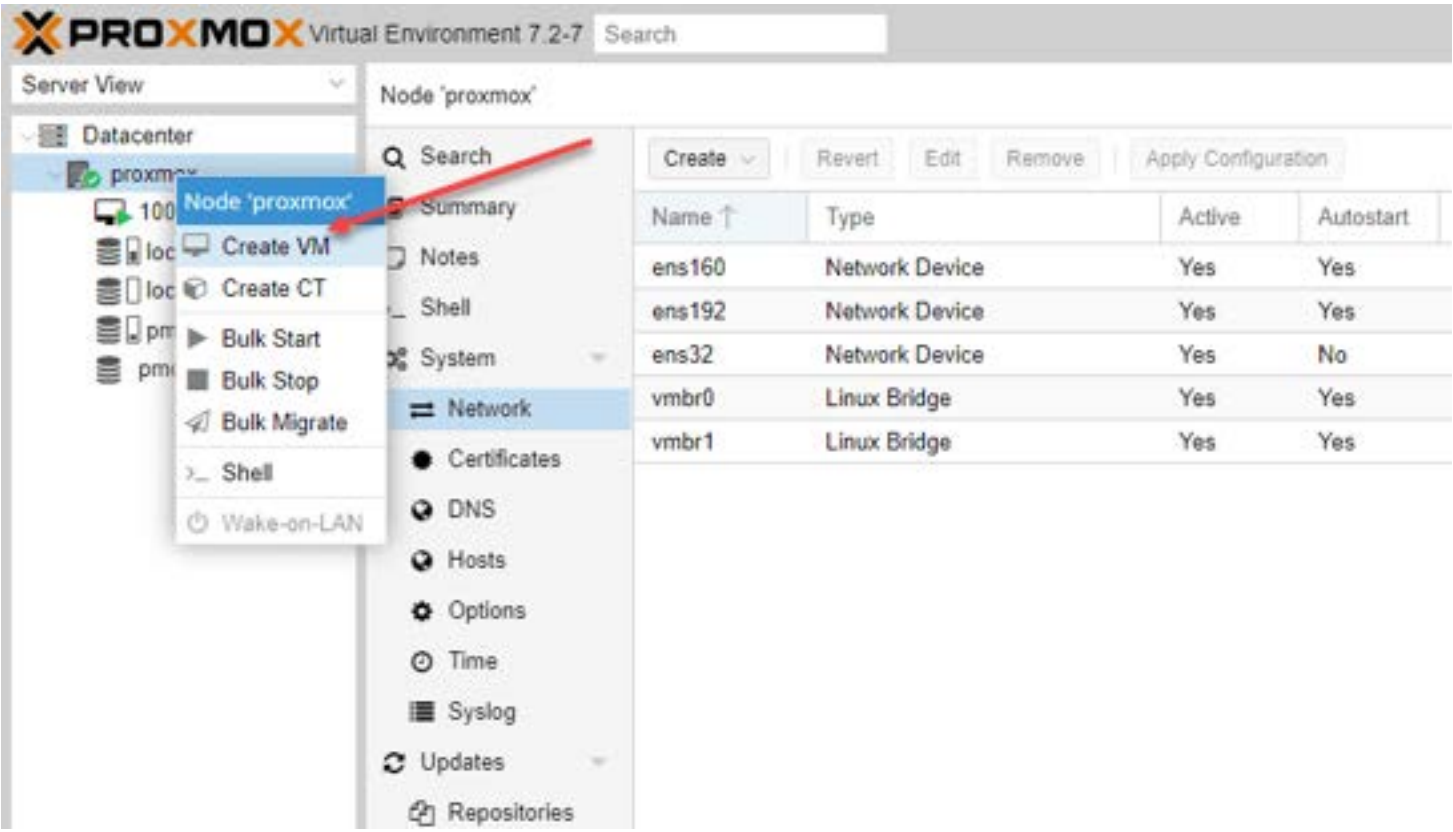


pfSense ISO image successfully uploaded to Proxmox VE server

Creating the pfSense VM in Proxmox VE

We first need to create the pfSense VM in Proxmox VE that will be used to install pfSense.

After you access Proxmox through port 8006, right-click your Proxmox VE server in the Proxmox web GUI and select Create VM.



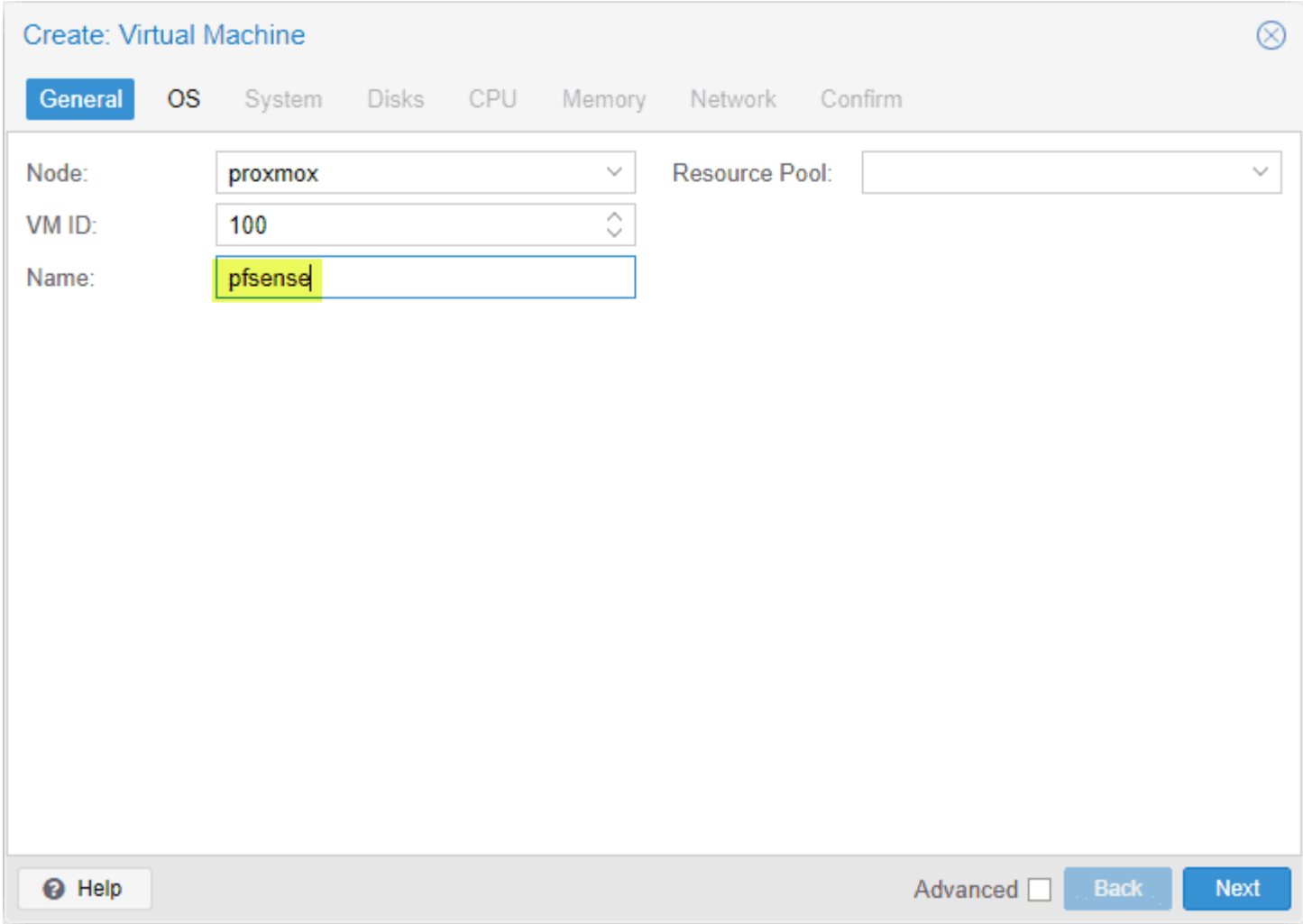
Create a new VM in Proxmox VE

Configuring the new pfSense VM

Note the following tabs and how they are configured with the new pfSense VM.

General tab Settings

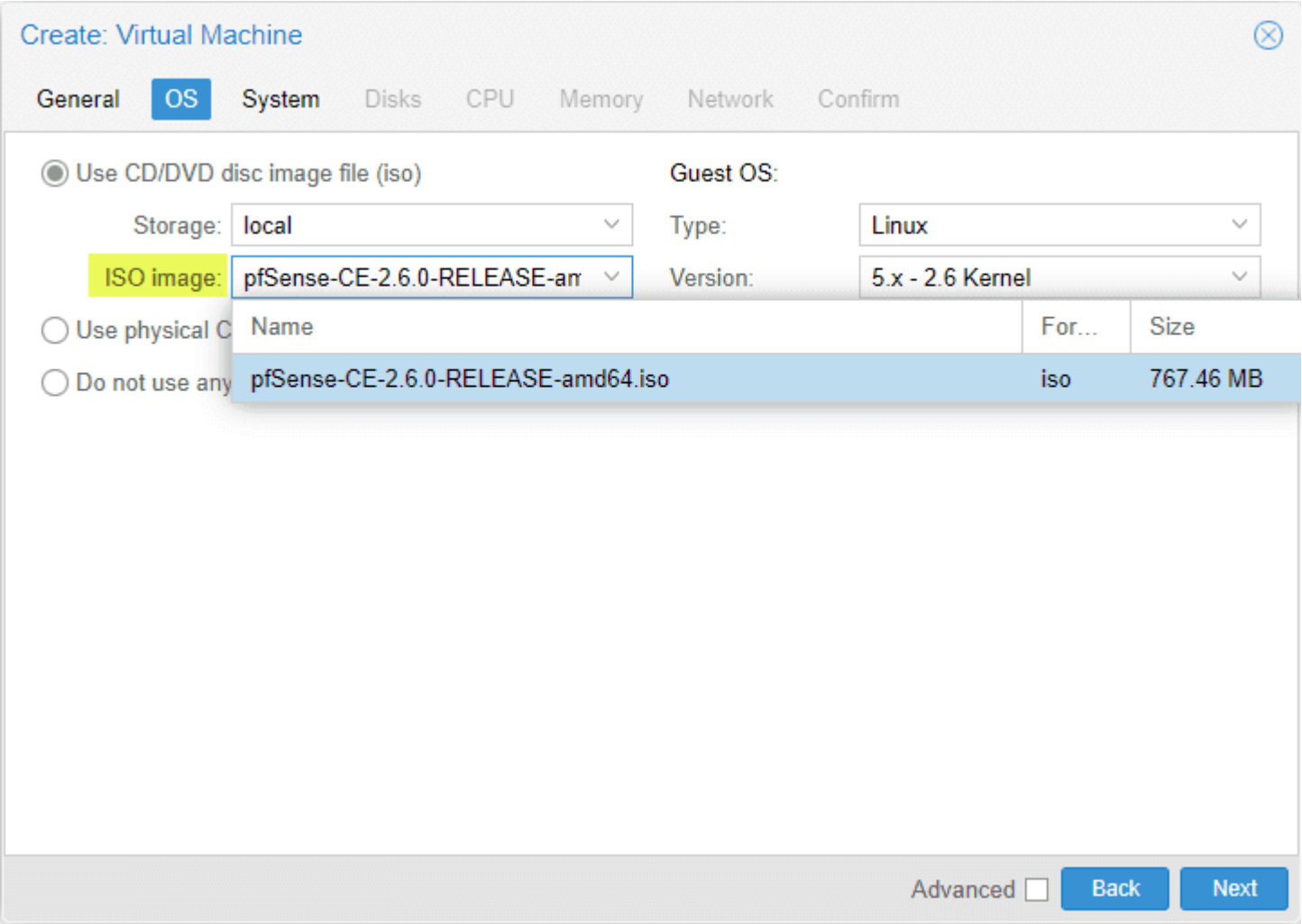
On the general tab, configure a name for the new pfSense VM.



Configure a name for the new pfSense VM

OS settings

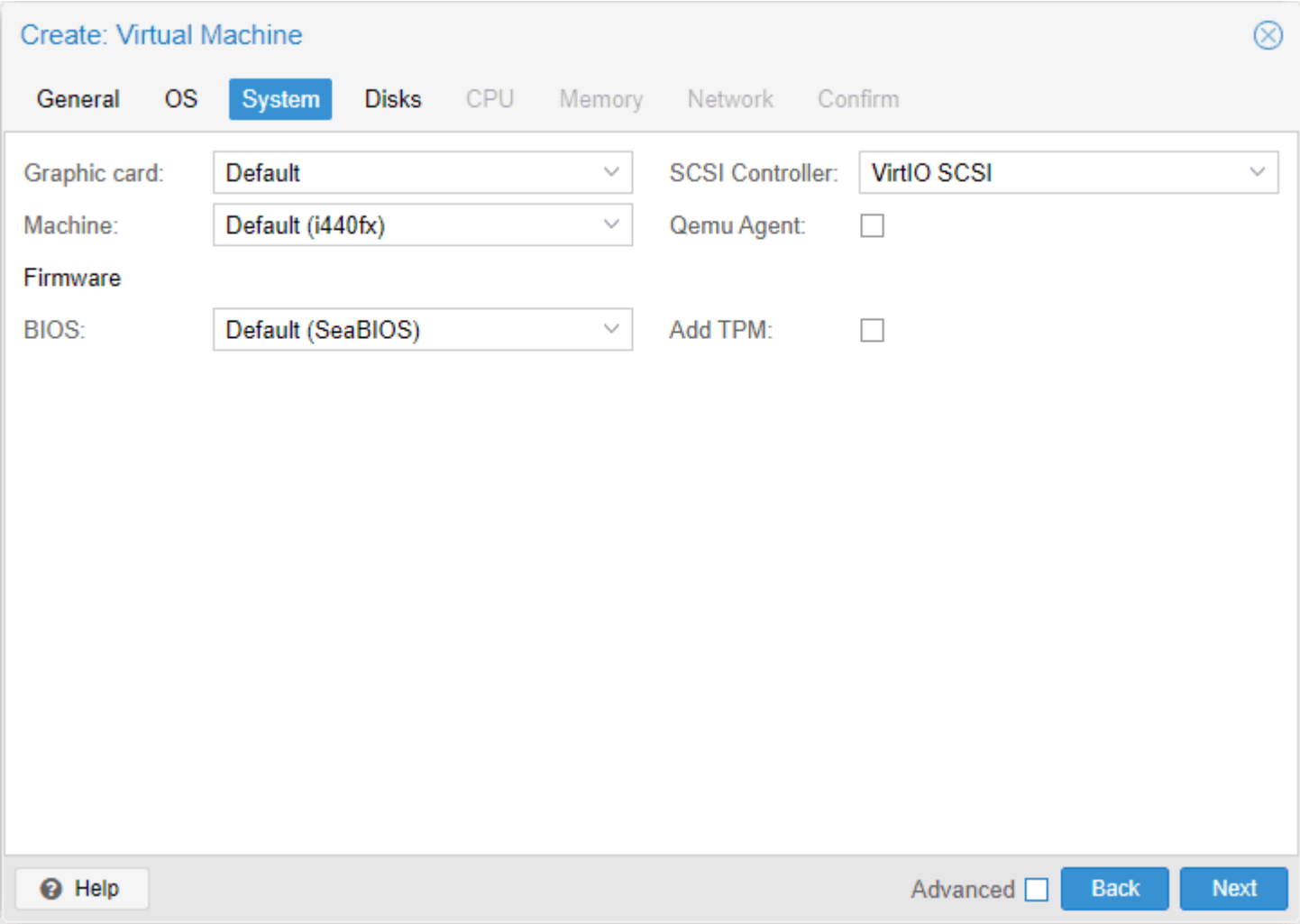
On the OS tab, here is where we select the ISO image that we uploaded earlier.



Select the pfSense ISO image

System tab

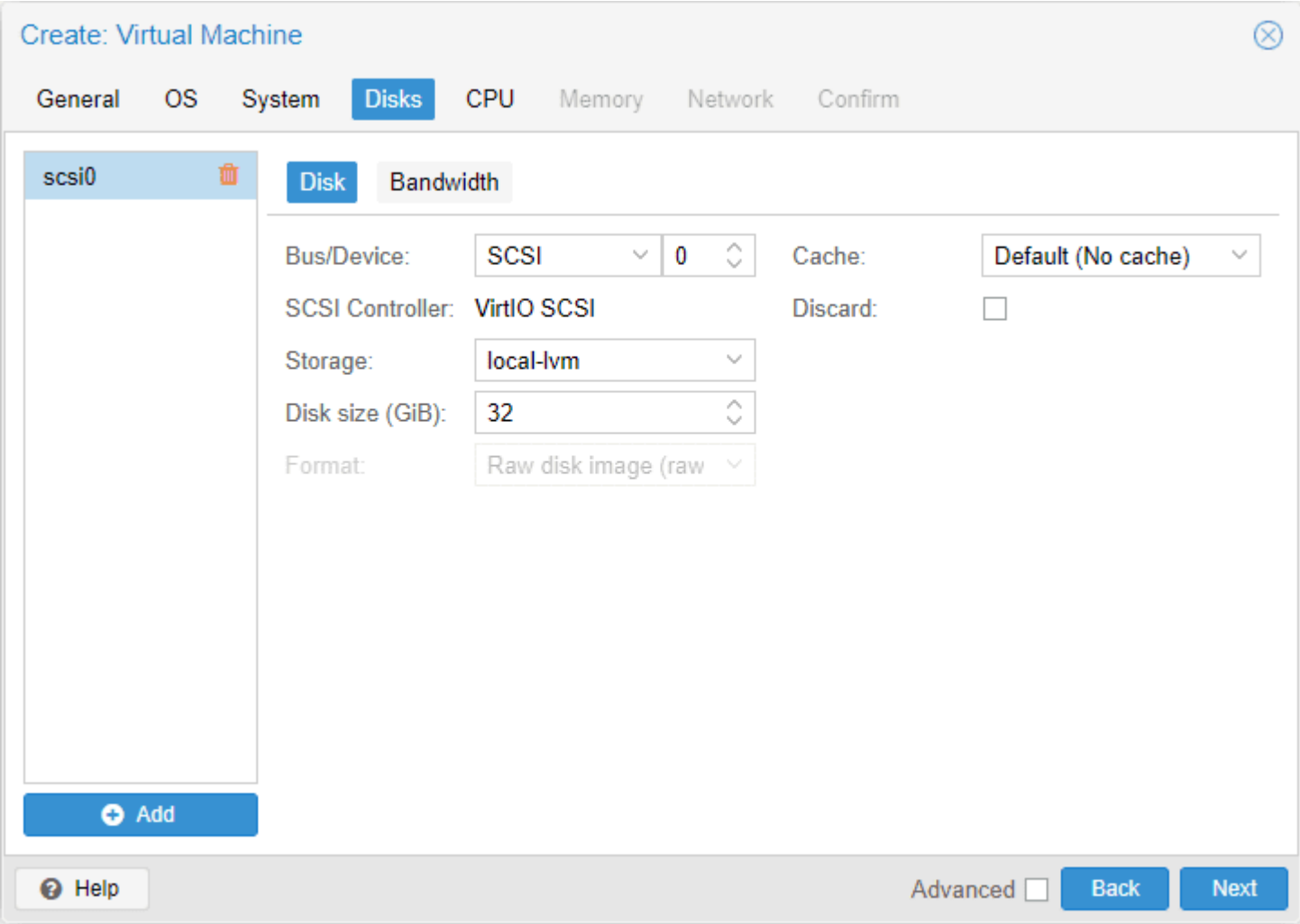
On the system tab, we can leave the default settings.



Accept the defaults on System

Disks

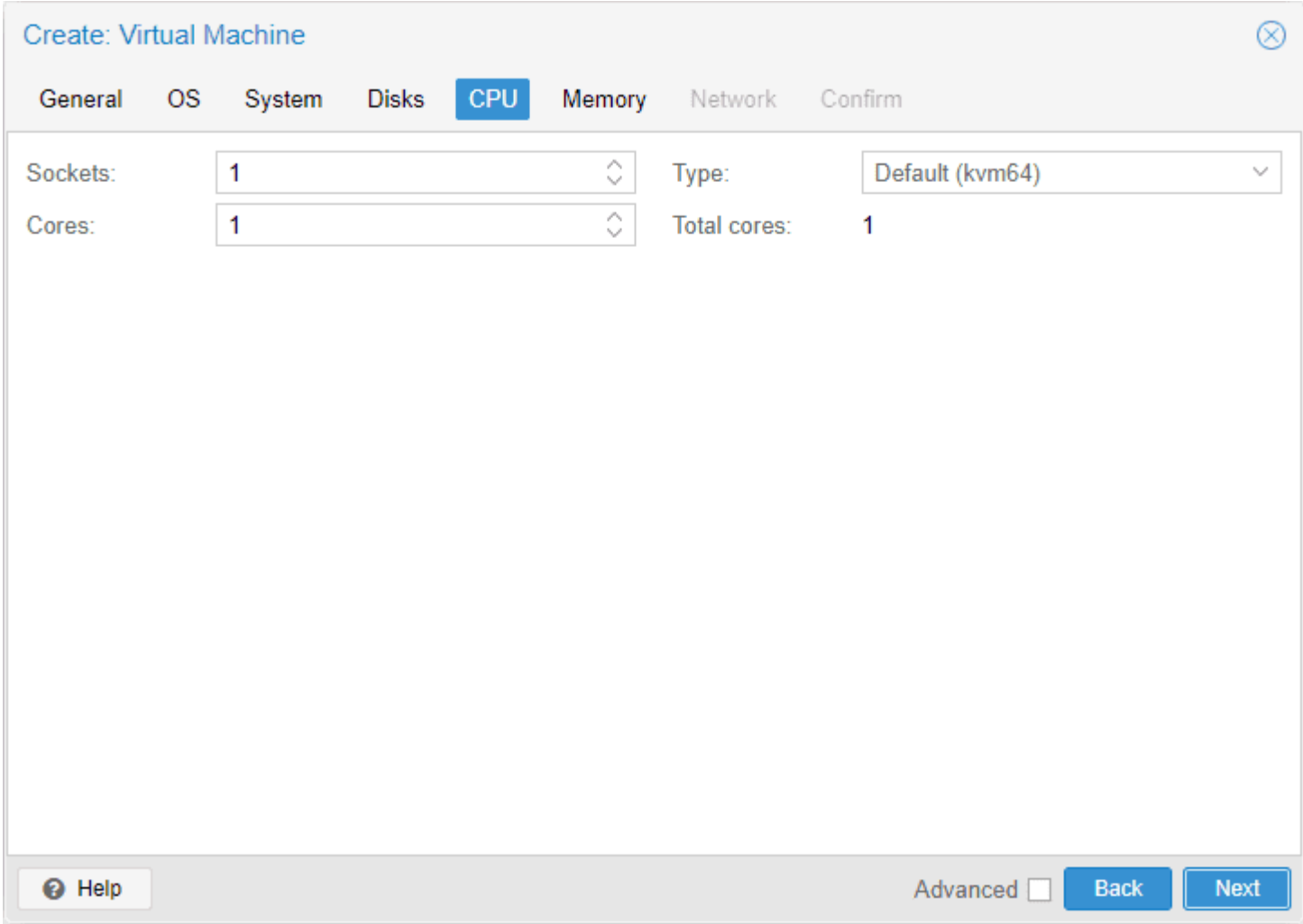
On the disk screen, you select where you want to install pfSense, the disk size, bus device information, etc.



Select the storage location for the pfSense VM in Proxmox

CPU tab

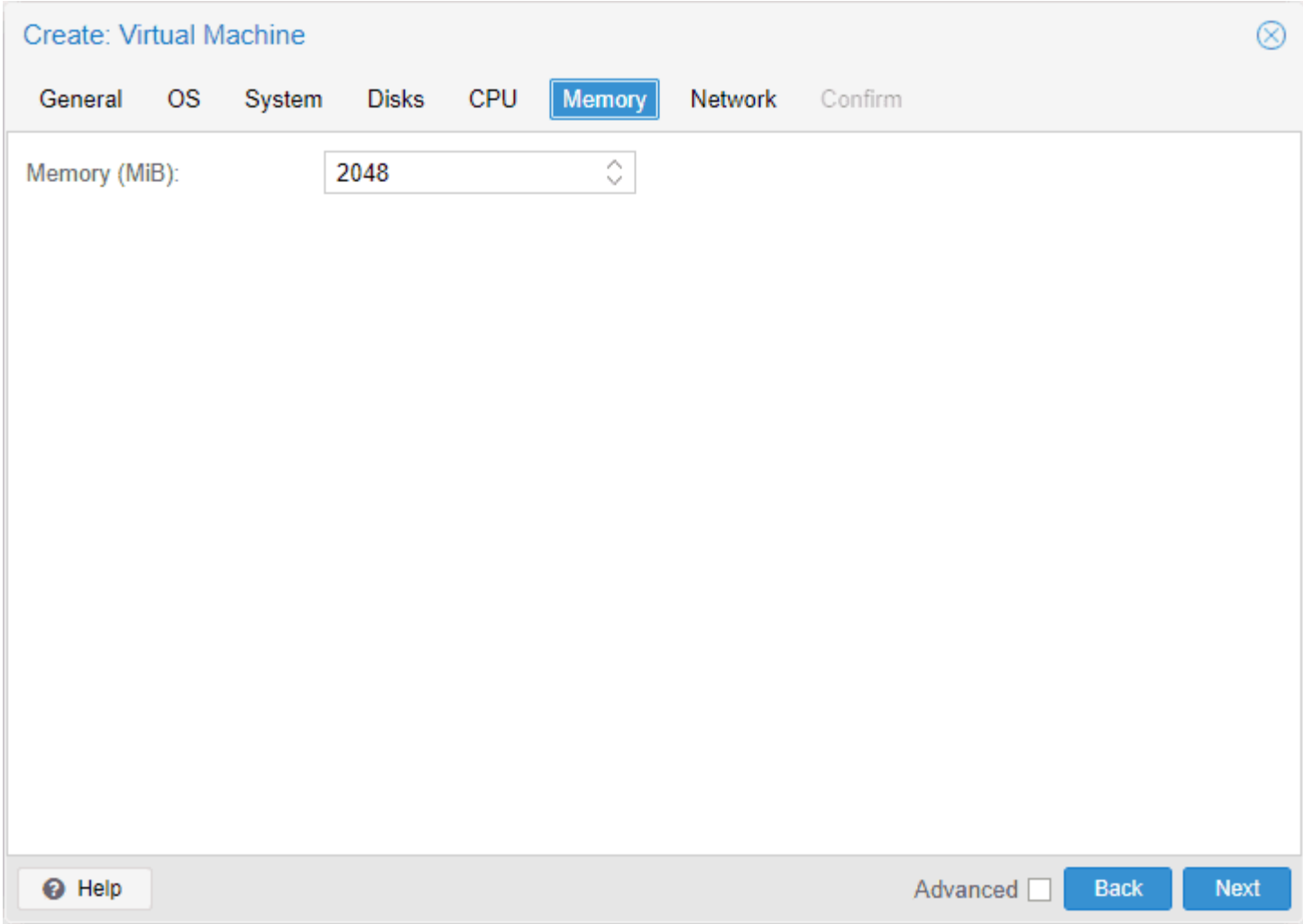
On the CPU tab, you can configure the number of CPU sockets and cores.



Configure CPU settings

Memory tab

On the memory tab, you configure how much memory you want to allocate to the pfSense VM.

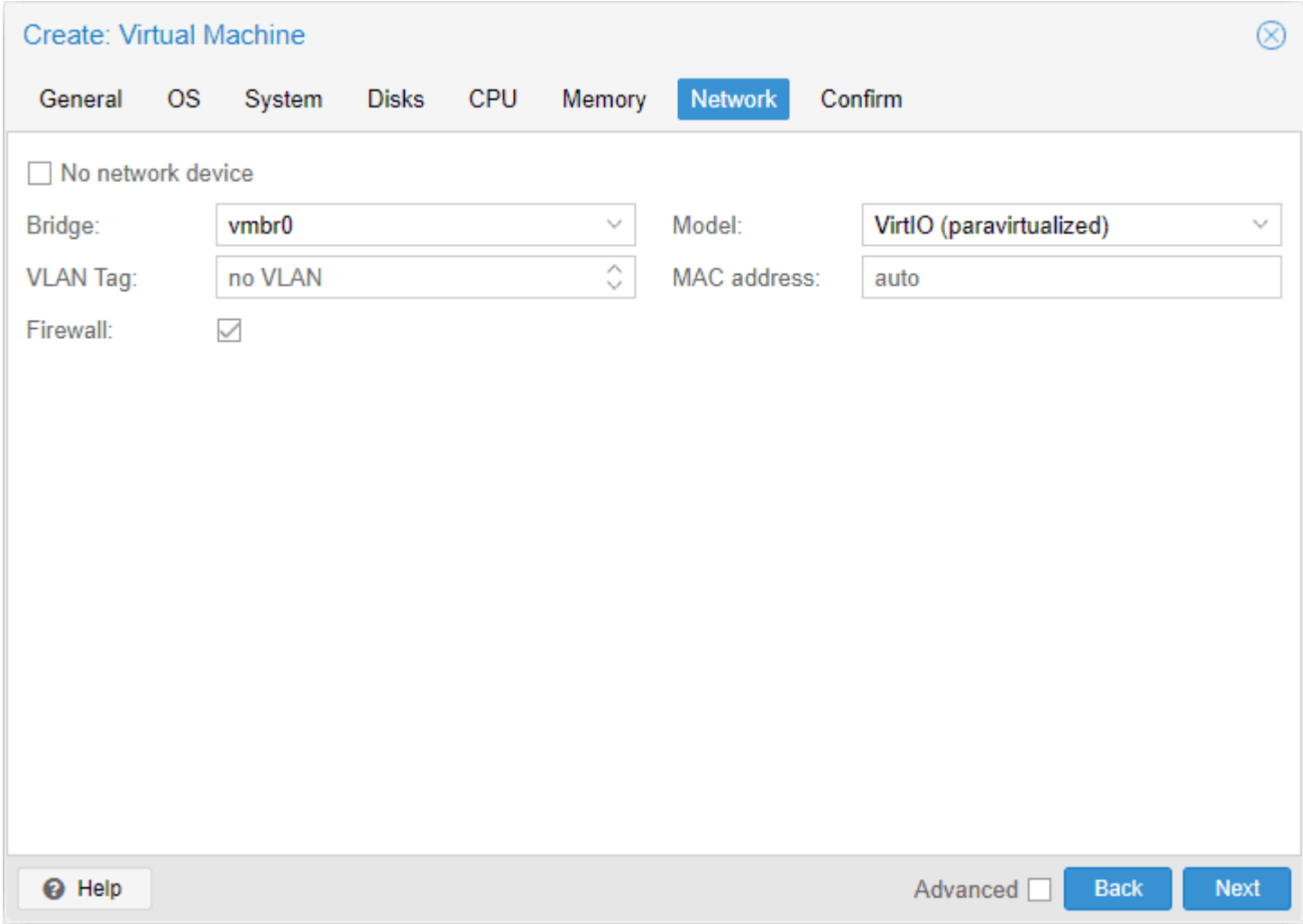


Configure memory settings for pfSense

Networking Tab

On the network tab, you configure the network interfaces you want to use for your pfSense VM running on your Proxmox host. There are differences to think about depending on whether you are running pfSense on physical hardware with physical interface ports or a virtual machine running pfSense.

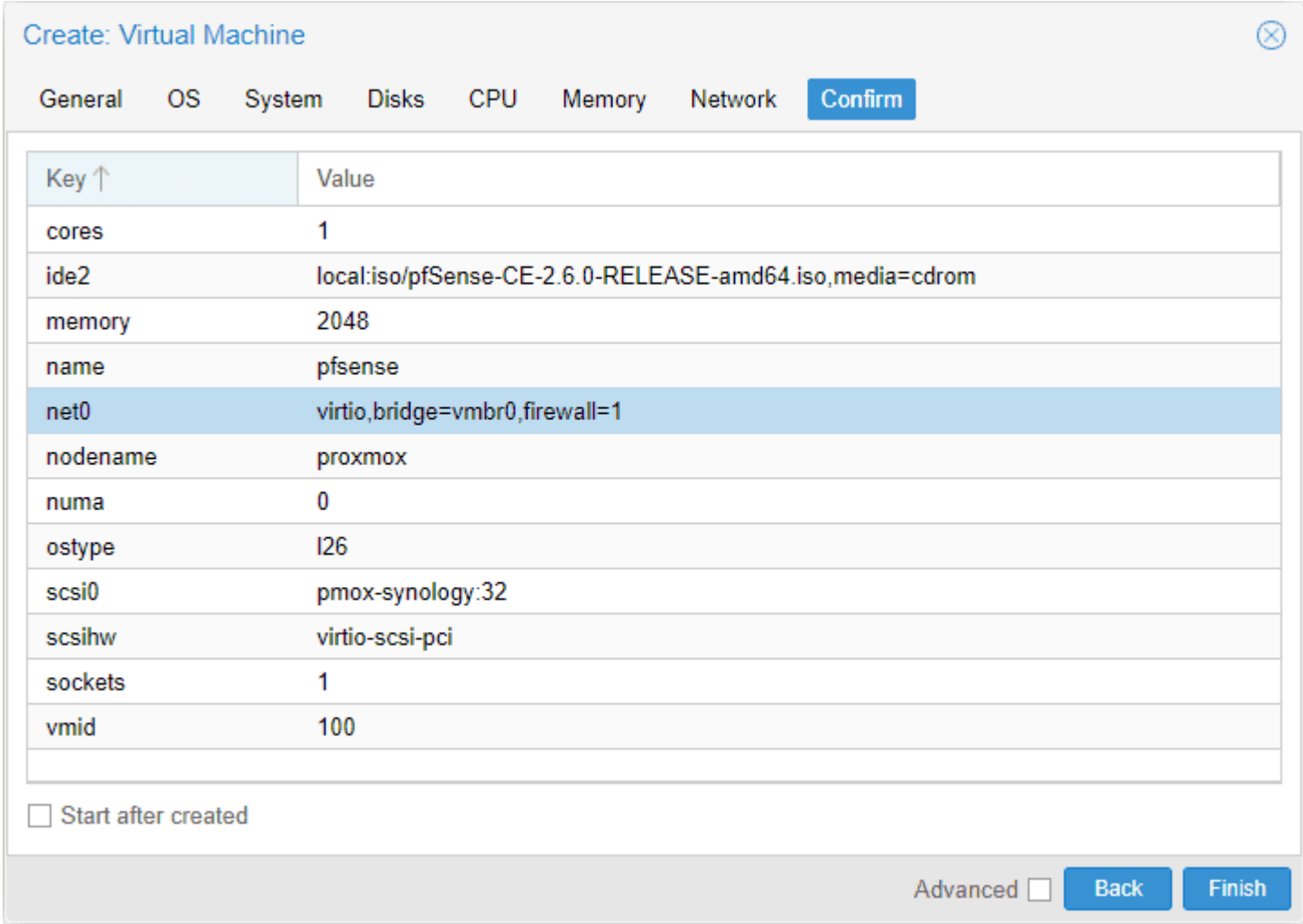
Here, on the creation screen, we can just accept the defaults and then we will change a couple of settings once we have the VM created. Note on the screen the settings you can configure, including bridge ports, VLAN tag, firewall, model, MAC address, etc.



Configure the network settings for the pfSense VM

Confirm tab

On the confirm tab, we can confirm the settings used to create VM for pfSense.



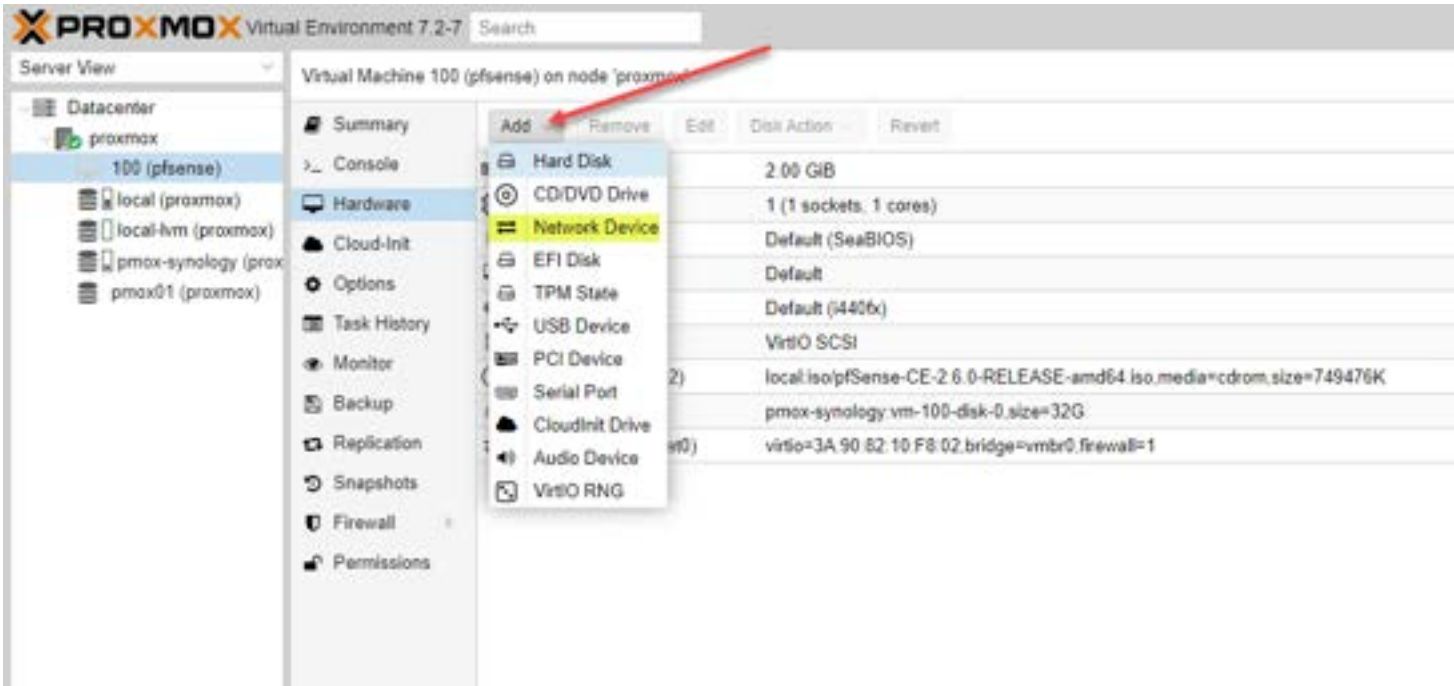
Confirm the configuration settings for the new pfSense VM

After you click create VM of the pfSense VM, this essentially creates the pfSense virtual machine so we can install pfSense as a guest OS on in the Proxmox box VM.

Changing a few settings on the pfSense VM

If you noticed on the network device screen above, it only configured one network device. However, for the pfSense VM to route traffic as expected, we need both a LAN port, or LAN interface, and a WAN port, or WAN interface.

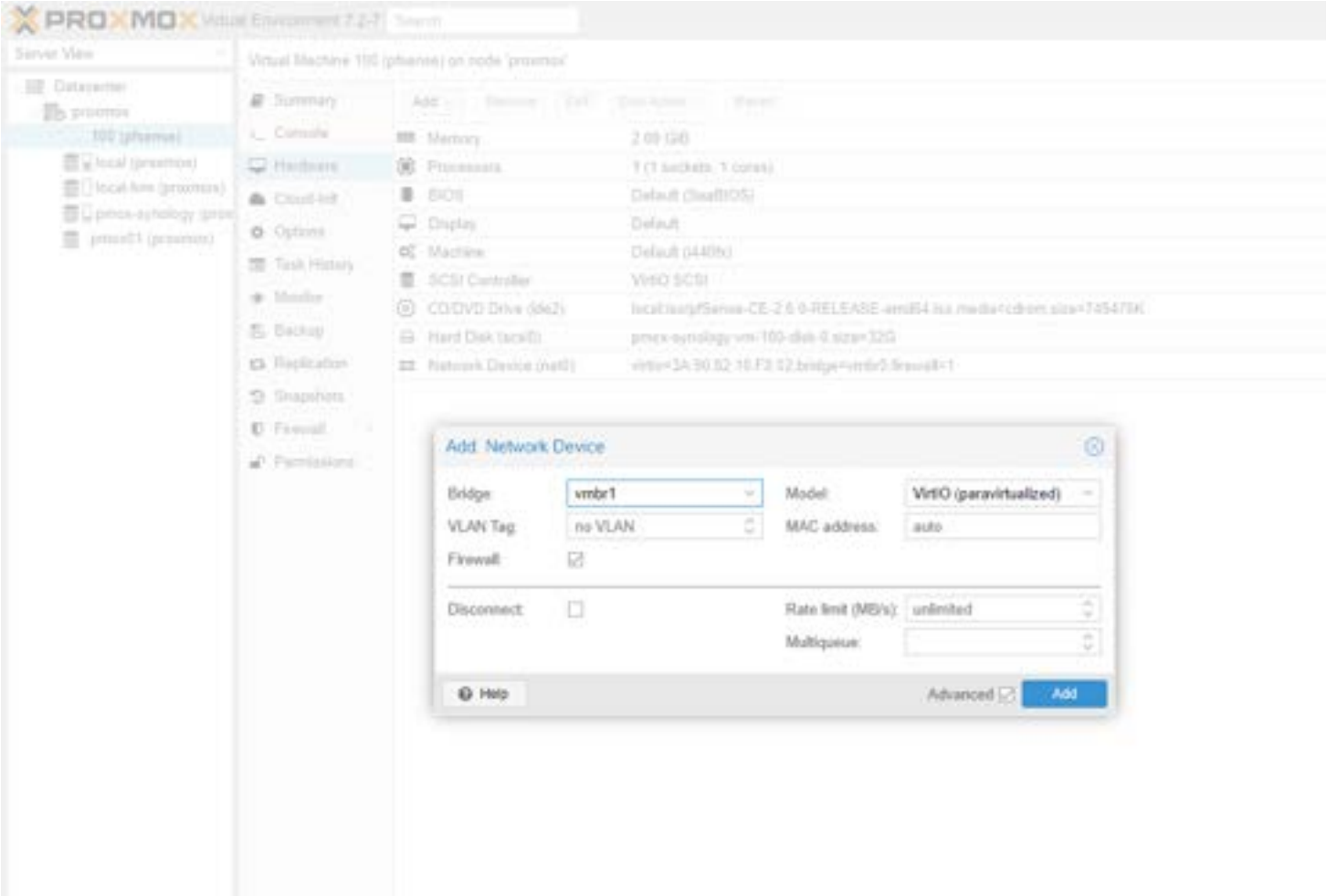
The WAN interface will house the WAN IP address that will provide connectivity from the outside inward for accessing internal resources and provide Internet connectivity. These WAN and LAN interface connections will allow successfully routing traffic as expected and benefiting from the pfSense firewall.



Add a new network adapter

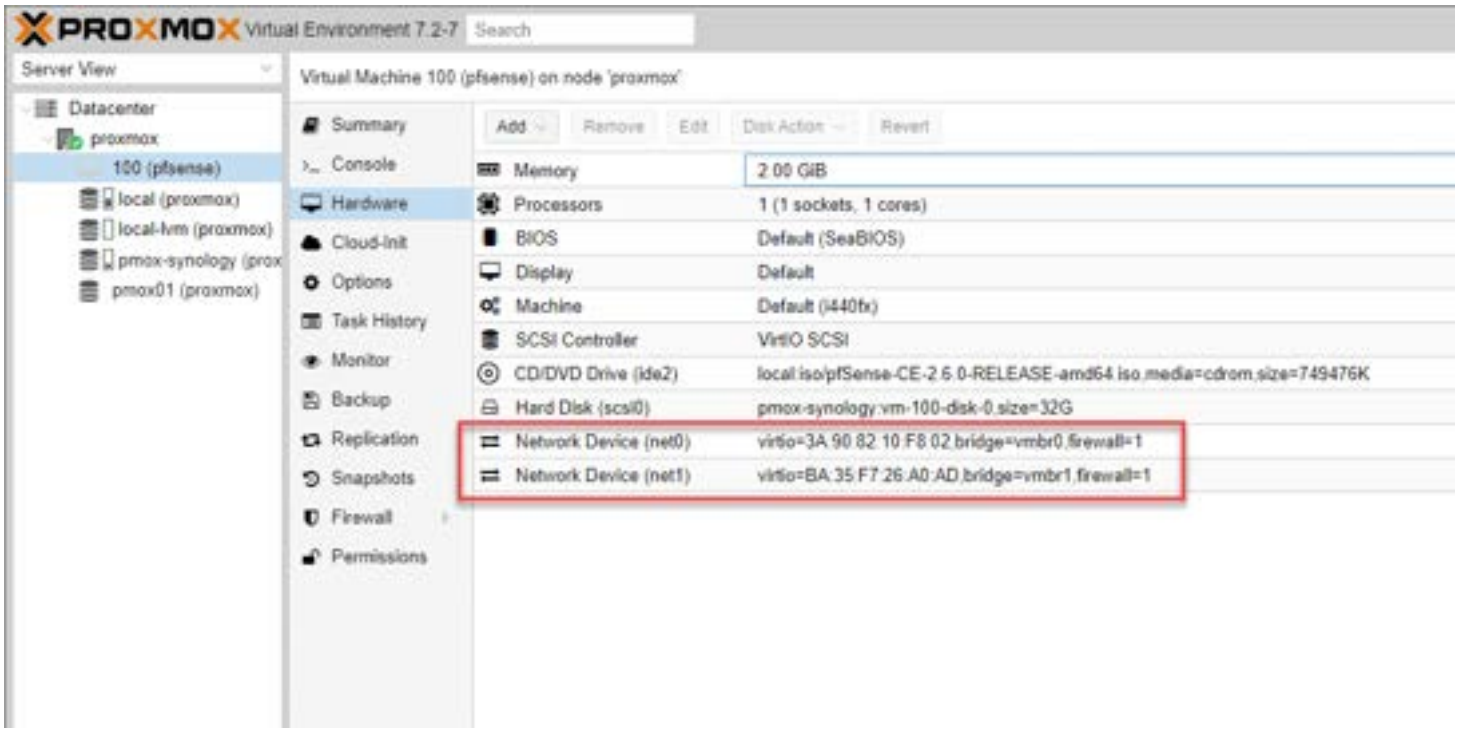
On the new interface, select the bridge ports, VLAN tag, and other settings for the second network adapter. By default, it will add virtIO interfaces when you add a new adapter. You may need to play around with this when adding. I had to go back and change my installation to Intel Pro 1000 adapters for it to work correctly in my nested lab.

I also added an additional network bridge where you can choose a new Linux bridge configuration.



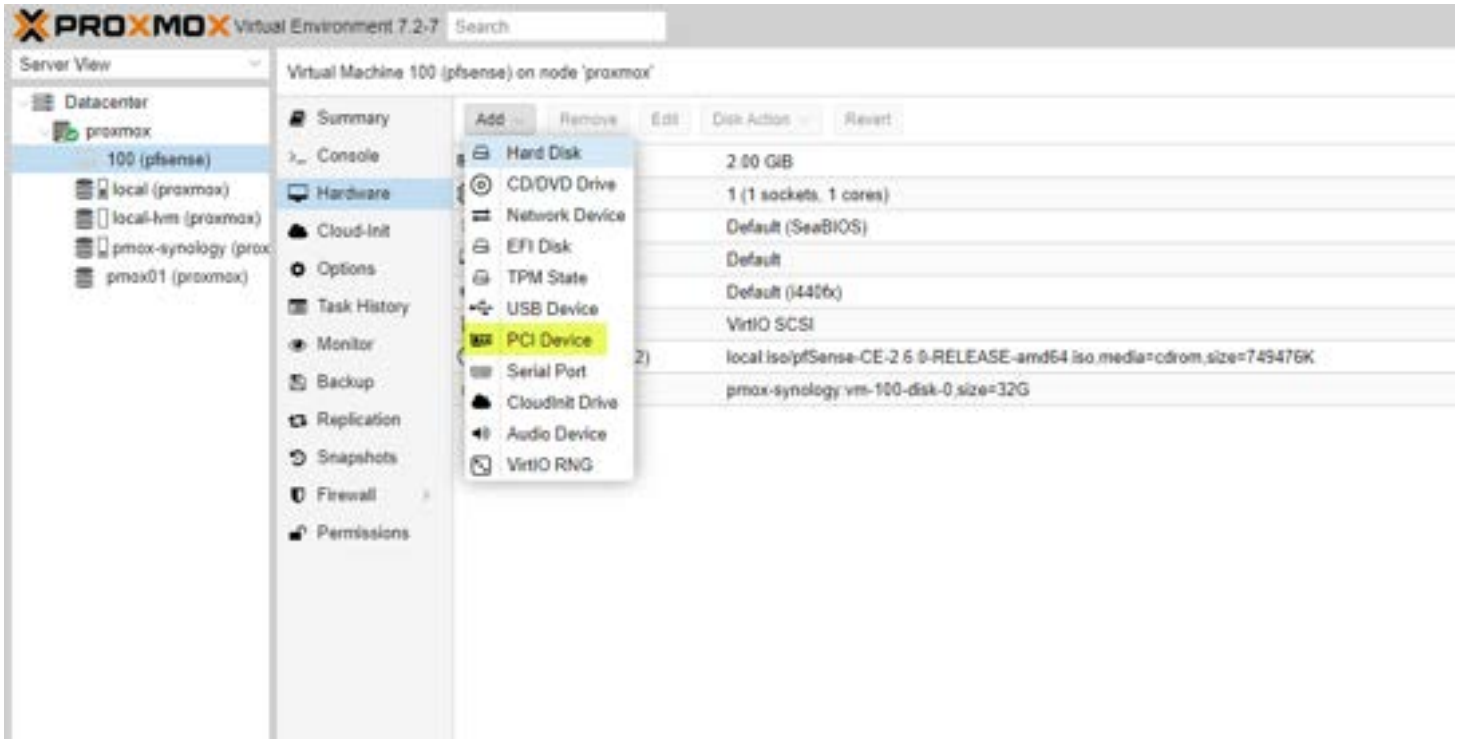
Add a new network adapter after creating the pfSense VM

After adding an additional network device, we now have two network devices configured with the pfSense VM.



Viewing the network adapters after adding to the pfSense VM

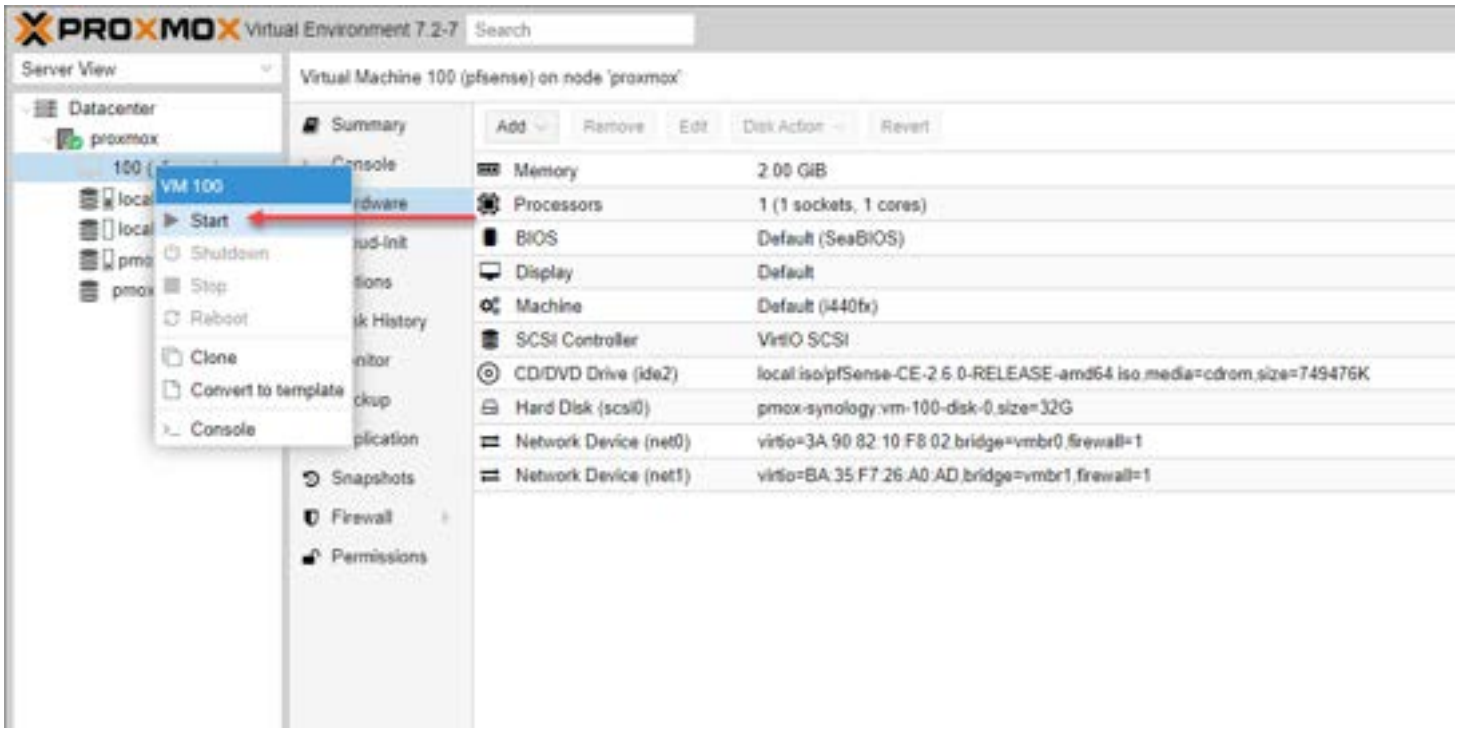
As a note, depending on what type of hardware you are running on top of for your Proxmox host, some may need to instead not add a network adapter but instead add a PCI device that is passed through to physical NICs.



Adding a new PCI device in Proxmox VE

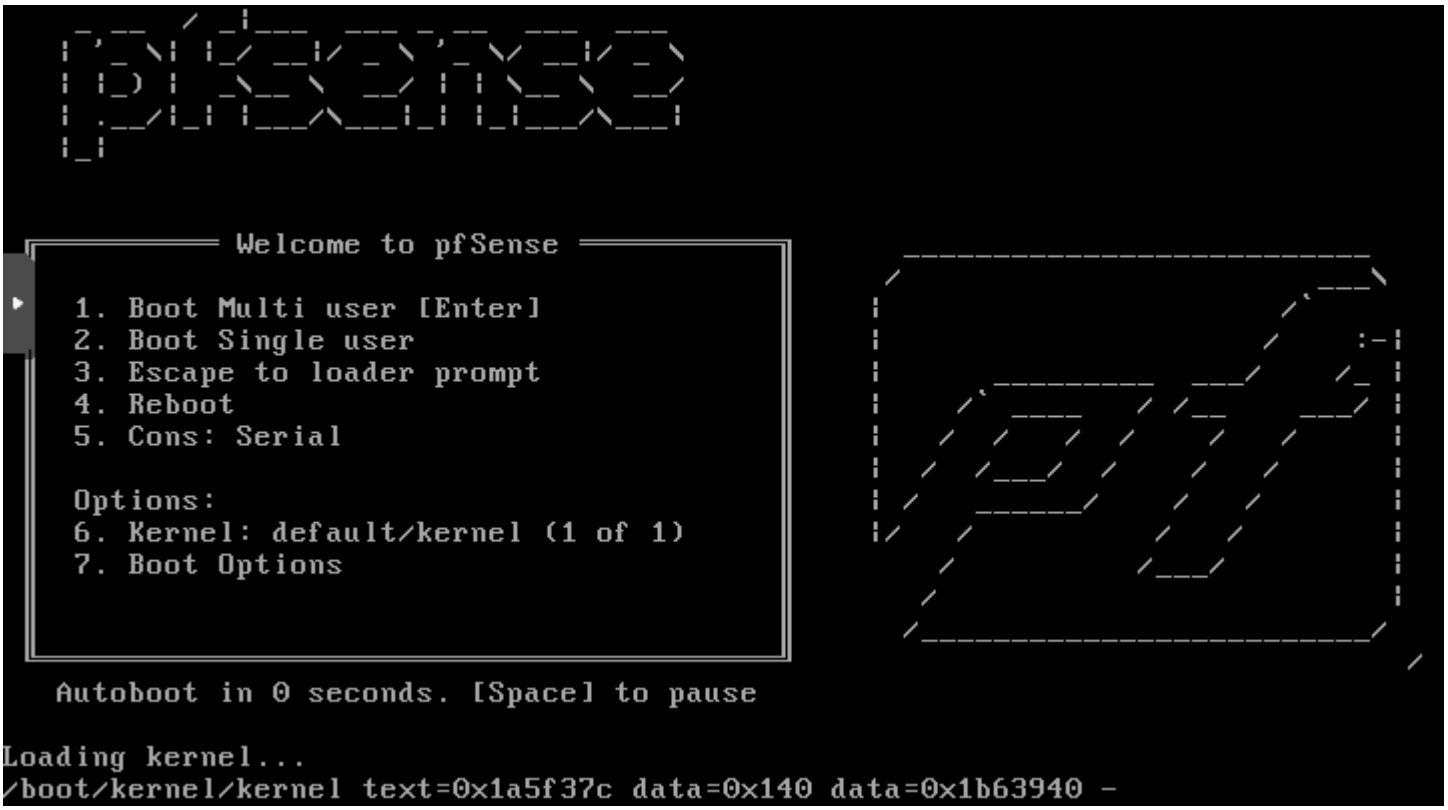
Install pfSense VM

Now we can actually install pfSense and configure the virtual machine appliance. Right-click the pfSense VM shown on your Proxmox host and select start.



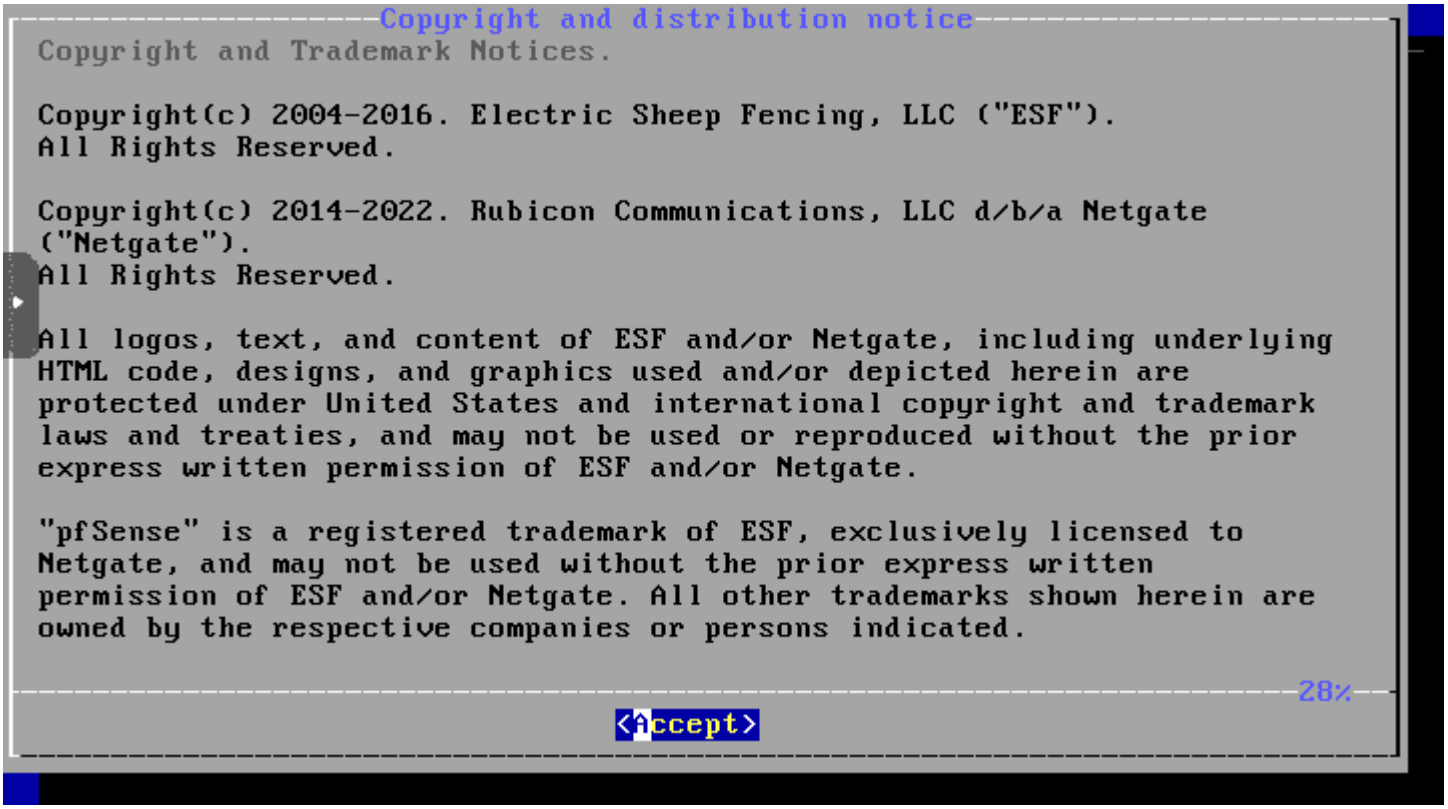
Power on the new pfSense virtual machine

After powering on and pfSense running as a VM, we can begin the process to run pfSense as an installed pfSense version.



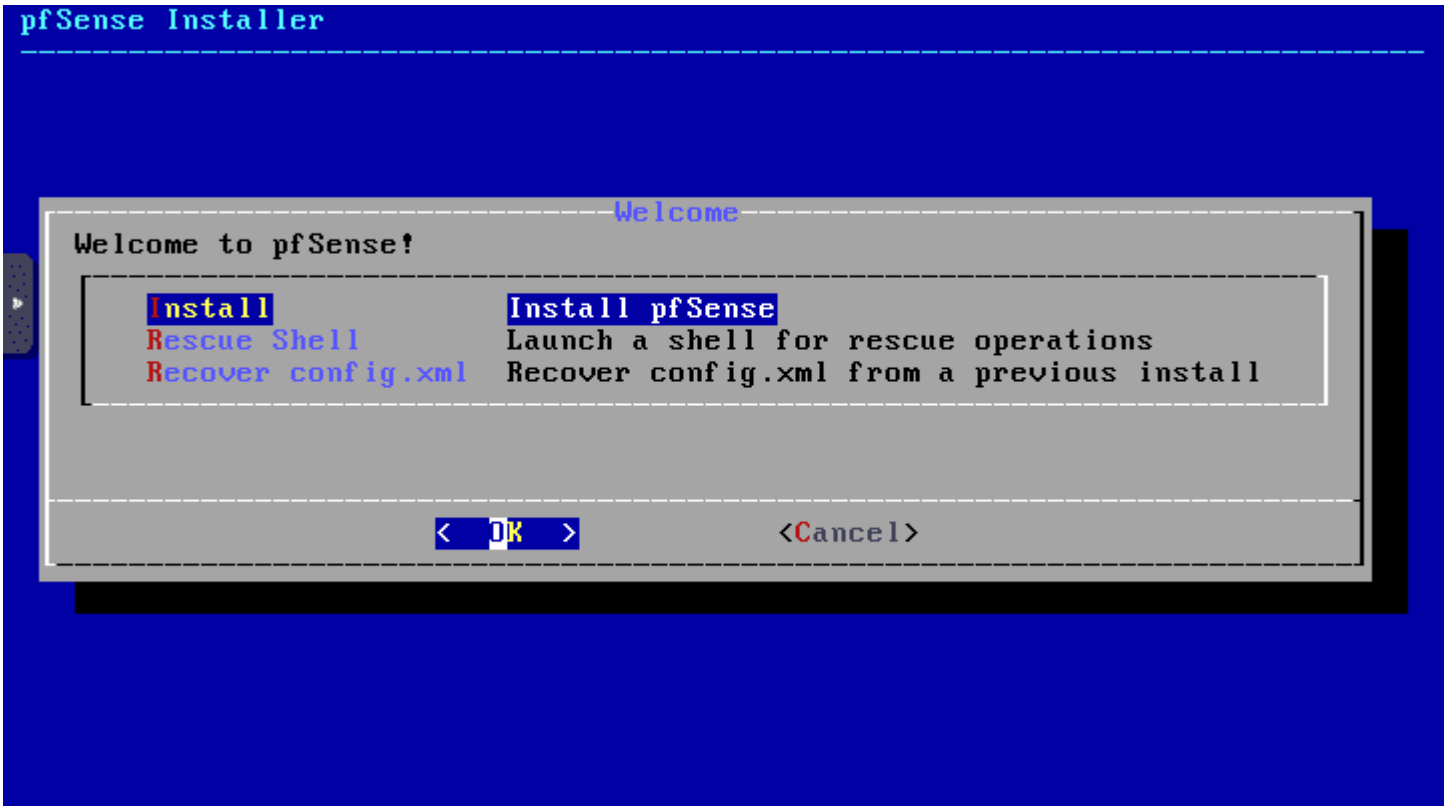
Boot screen of the pfSense VM running in Proxmox VE

This begins the “text” install pfsense VM process. Accept the EULA displayed.



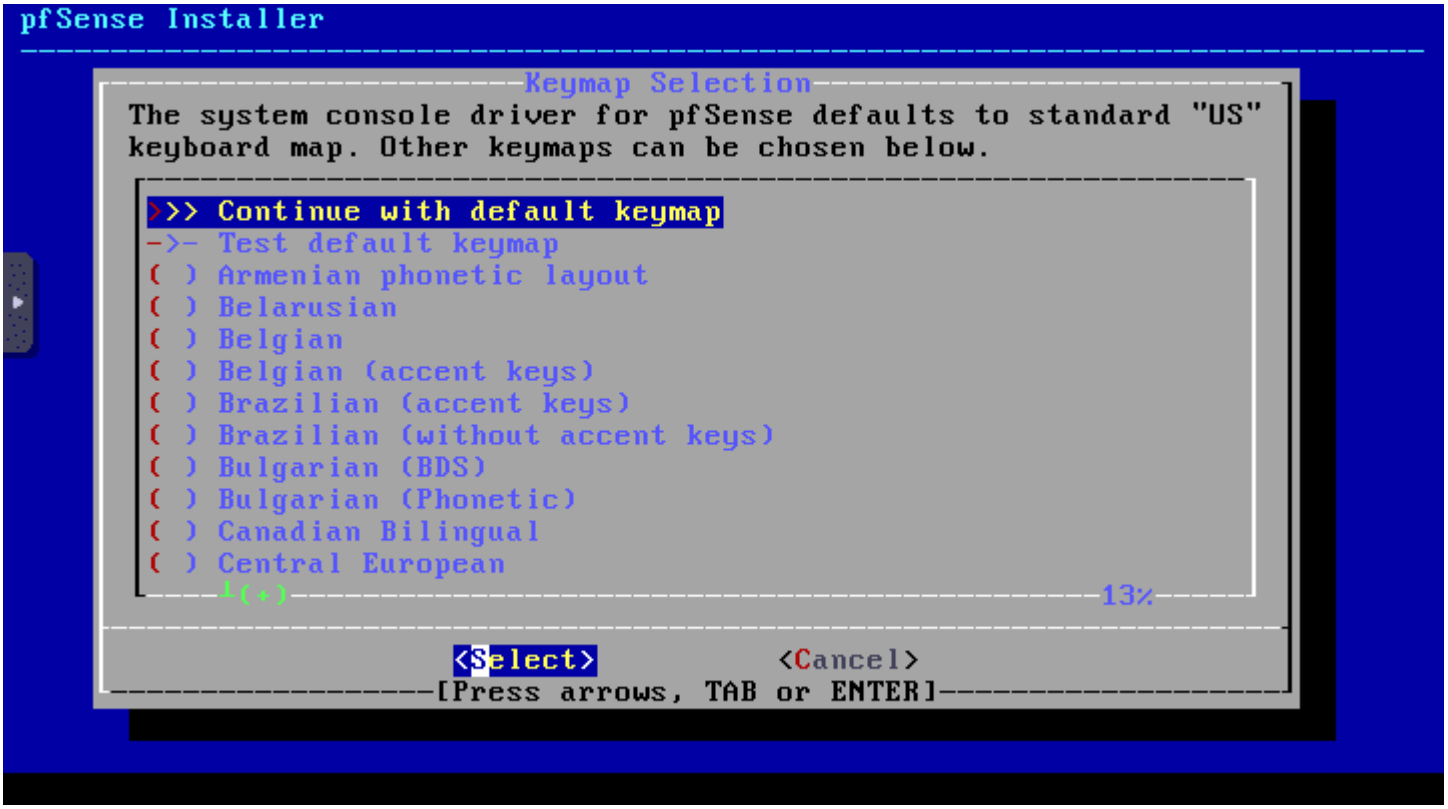
Accept the EULA agreement

Choose to install pfSense on the next screen.



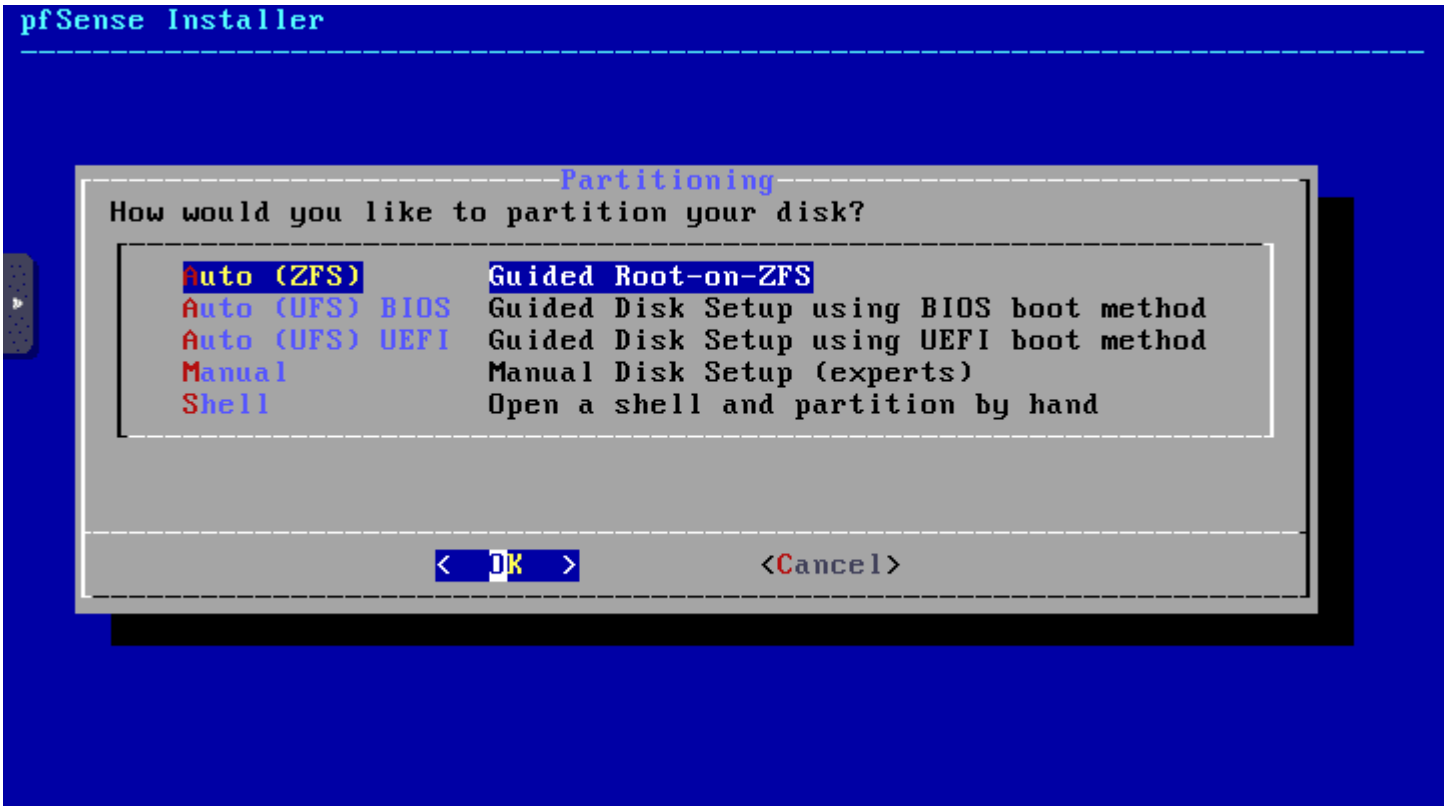
Selecting the Install option on the text installer screen

Continue with the default keymap for the keyboard layout.



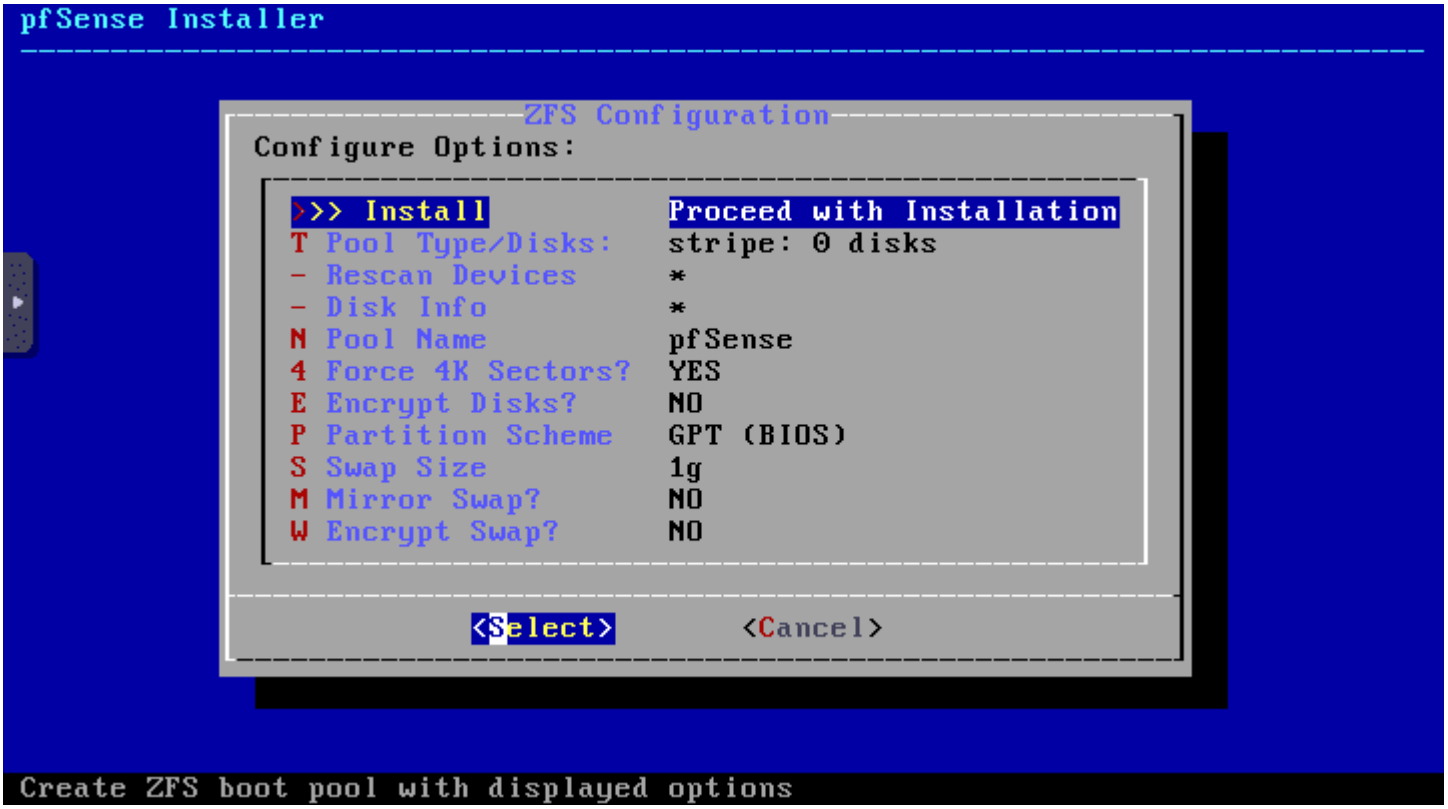
Configuring the default keymap

Choose to configure the partitioning unless you need a custom layout Automatically. Here I am choosing ZFS configuration.



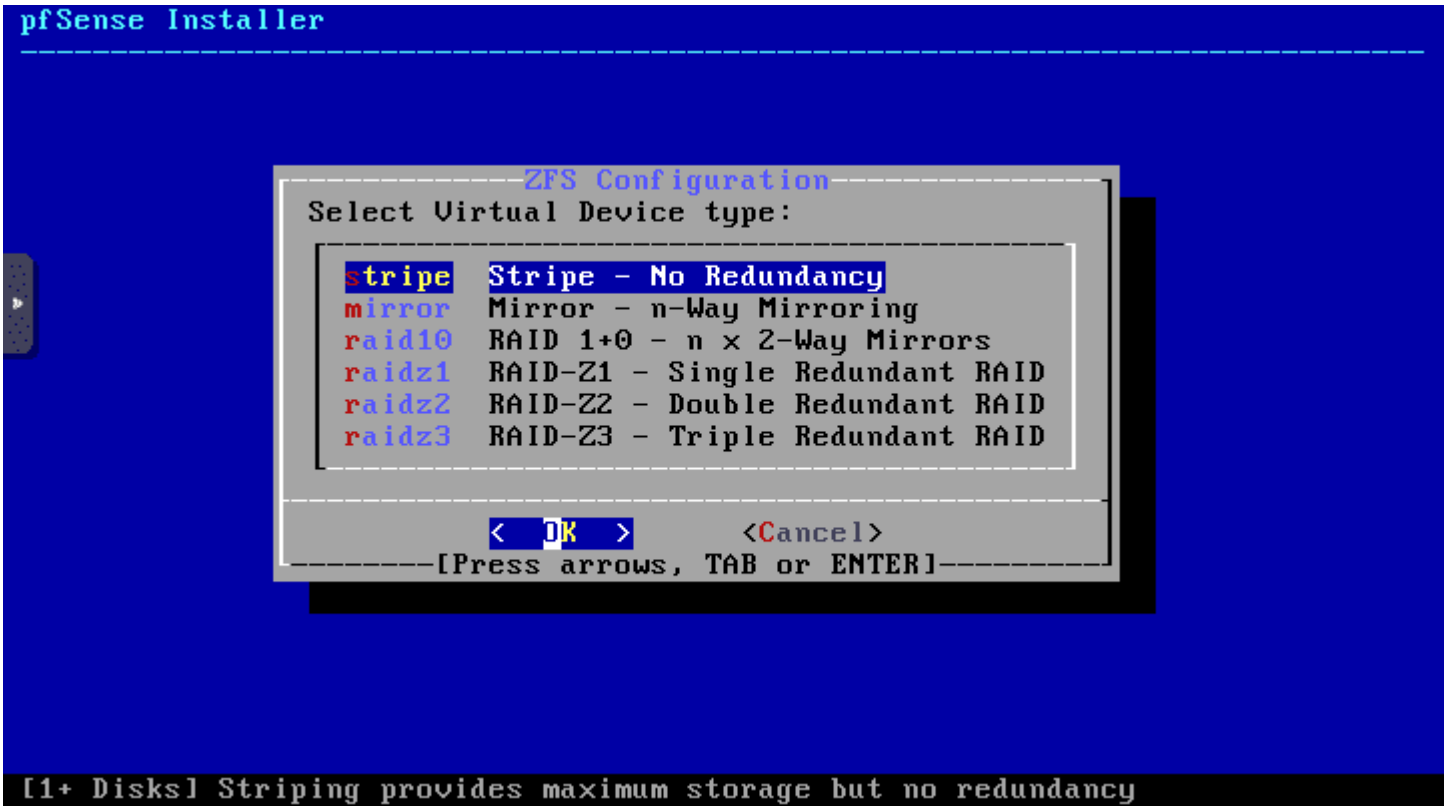
Selecting how you would like to partition your disk in Proxmox VE

Proceed with the install pfSense process.



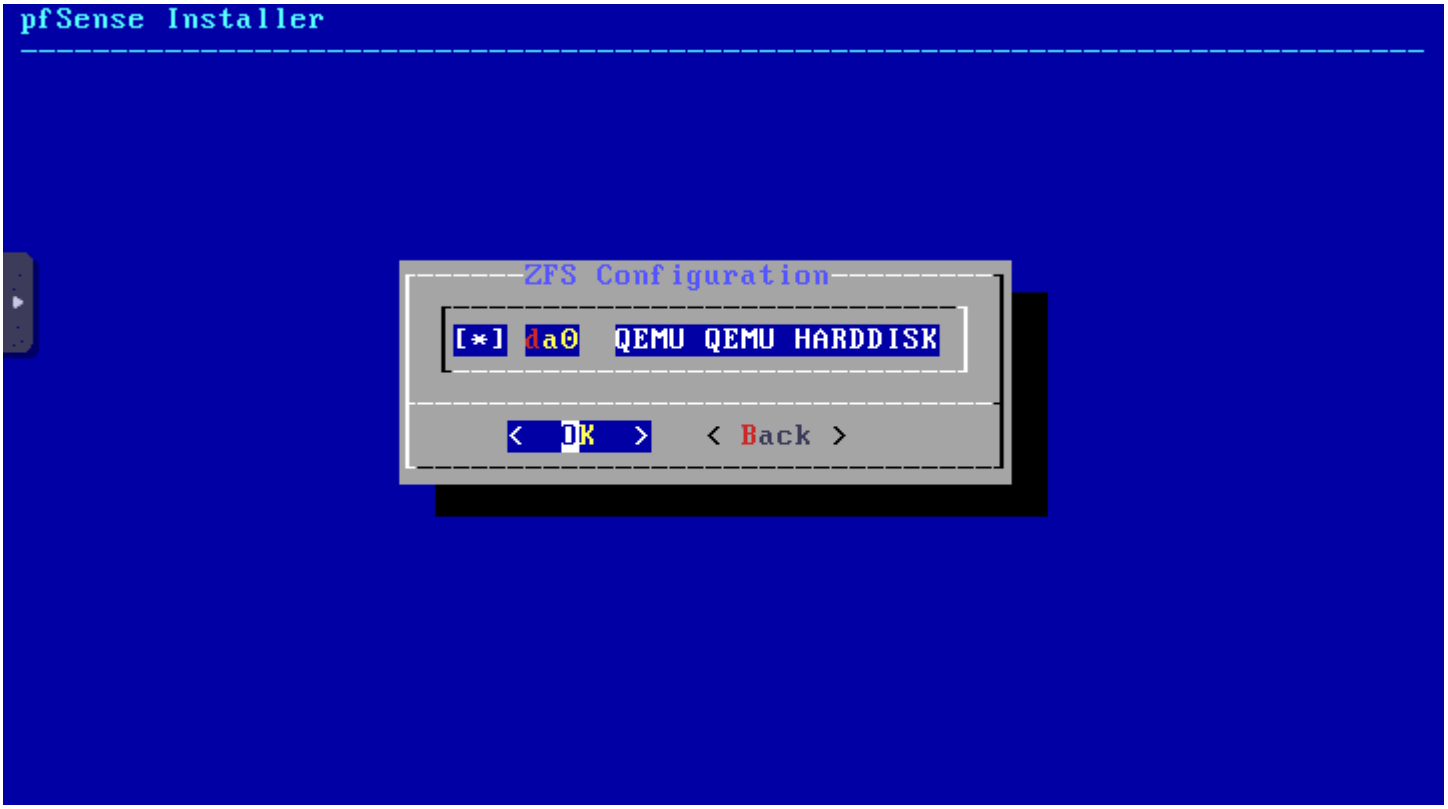
Confirming to proceed with the installation

Choose the virtual device type. Here I am selecting the Stripe no redundancy.



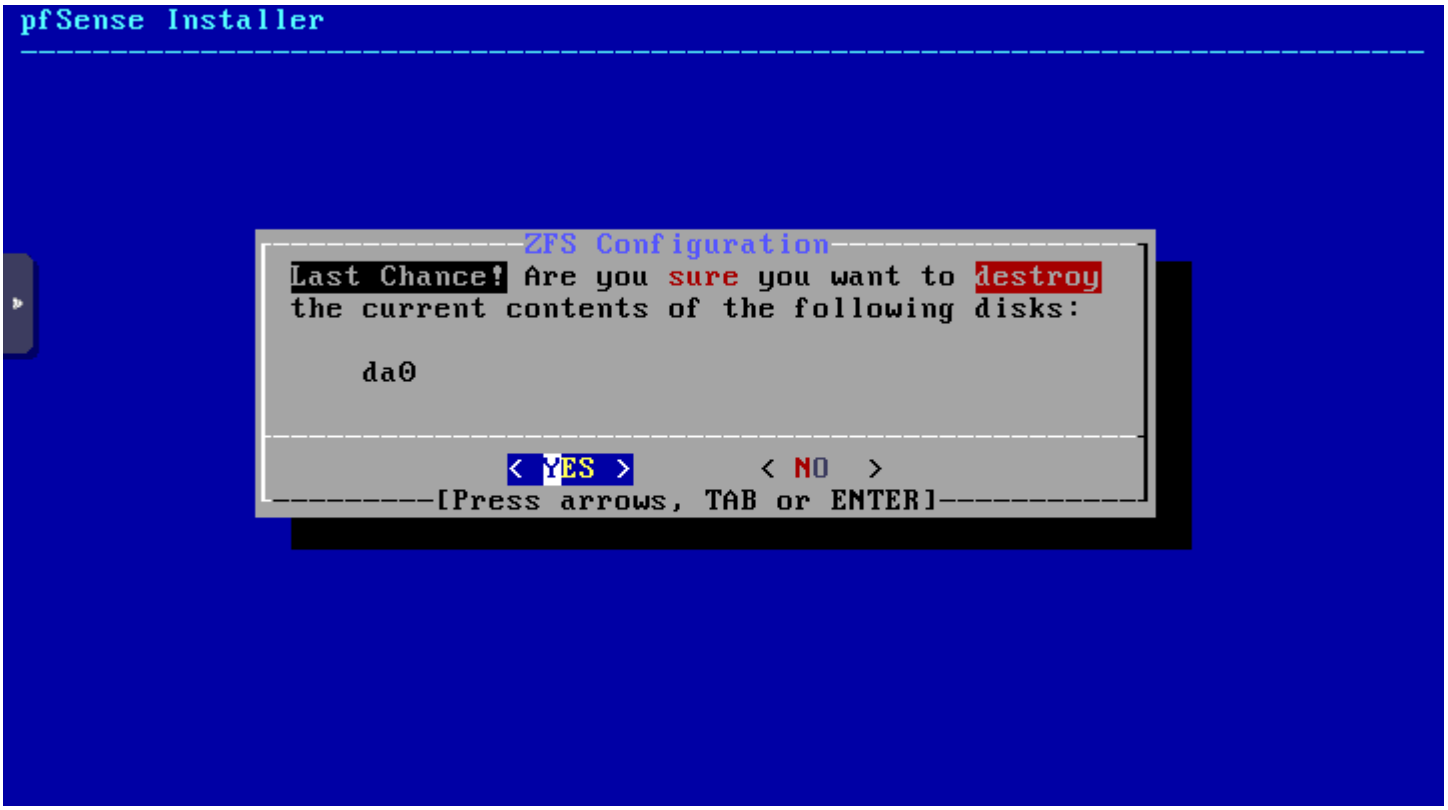
Select the virtual disk type

On the ZFS configuration screen, click OK.



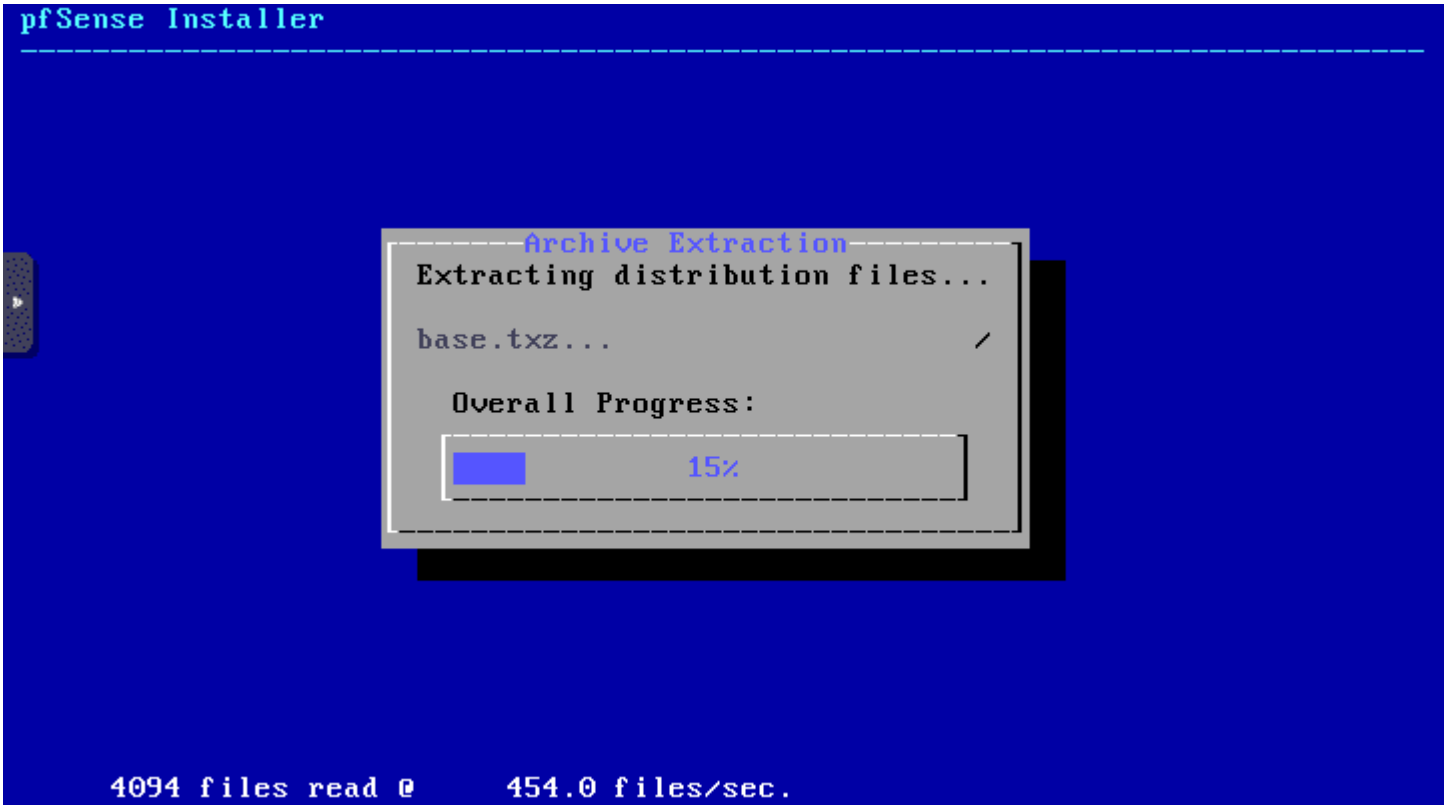
Confirm your ZFS configuration

Click Yes on the ZFS configuration screen.



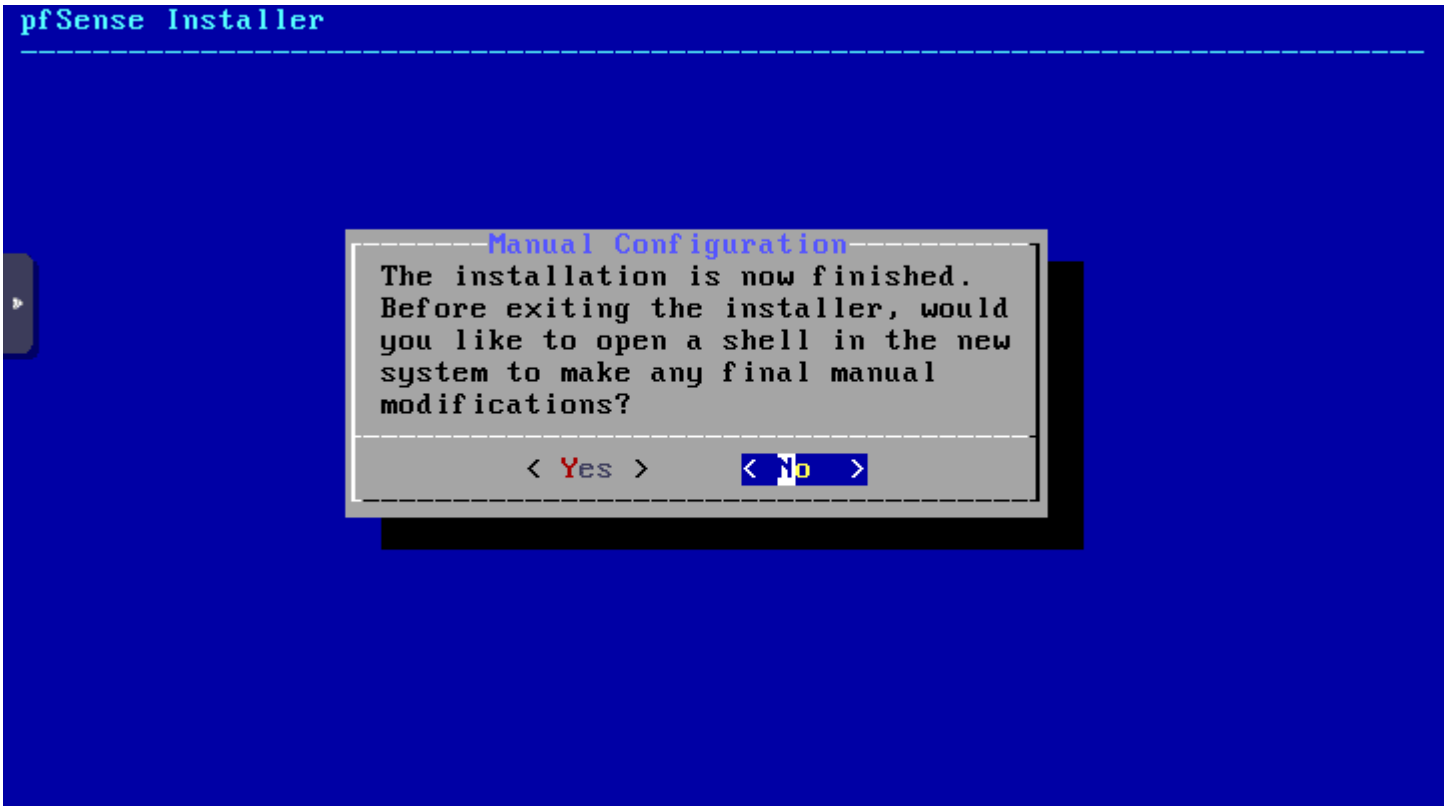
Confirming you want to format the disk and destroy data

The install pfSense process begins.



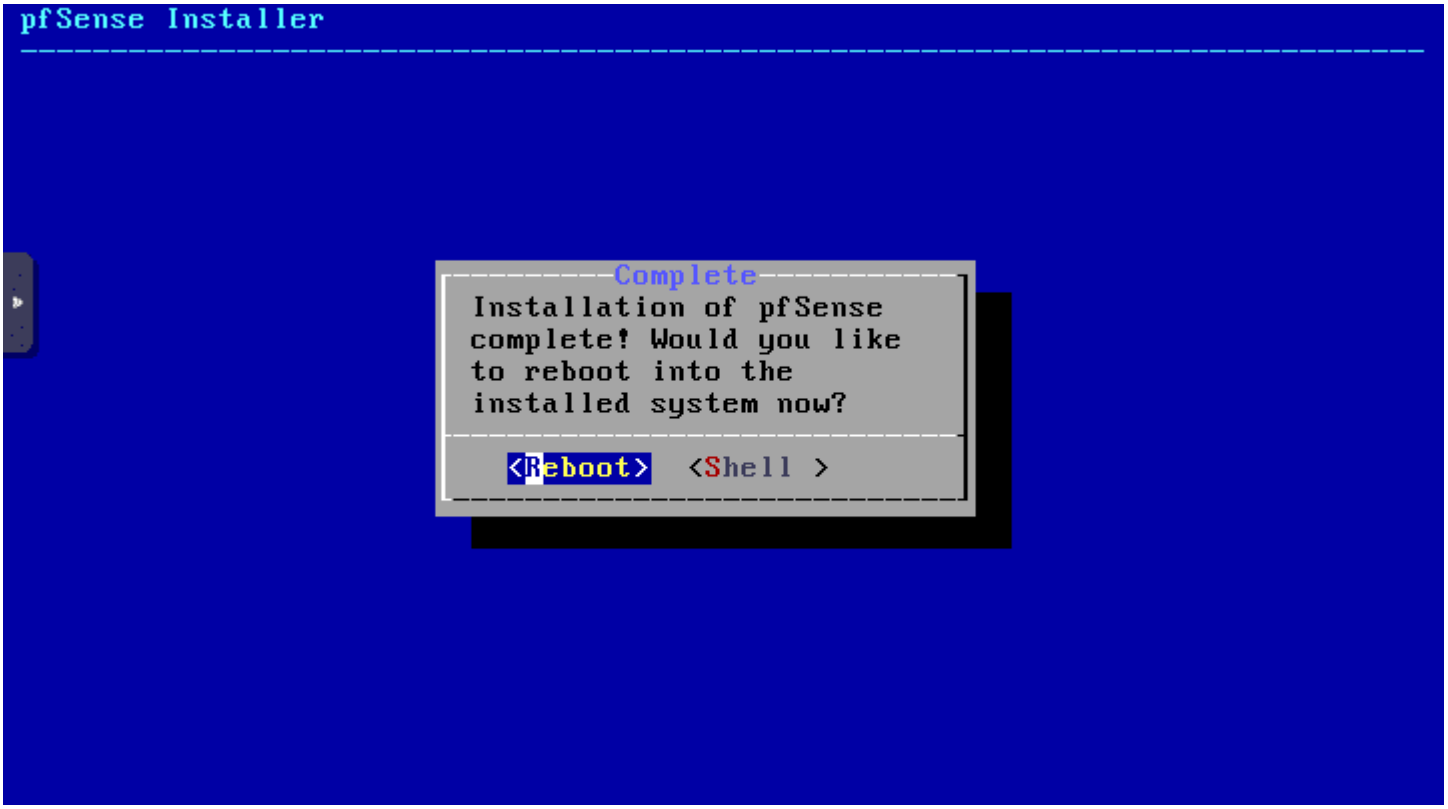
pfSense installation on Proxmox begins

You will be asked if you have any manual configuration you want to perform. If not, select No.



Installation is finished and choosing no custom modifications

The installation is complete. Reboot your pfSense VM.



Choosing to reboot after the installation

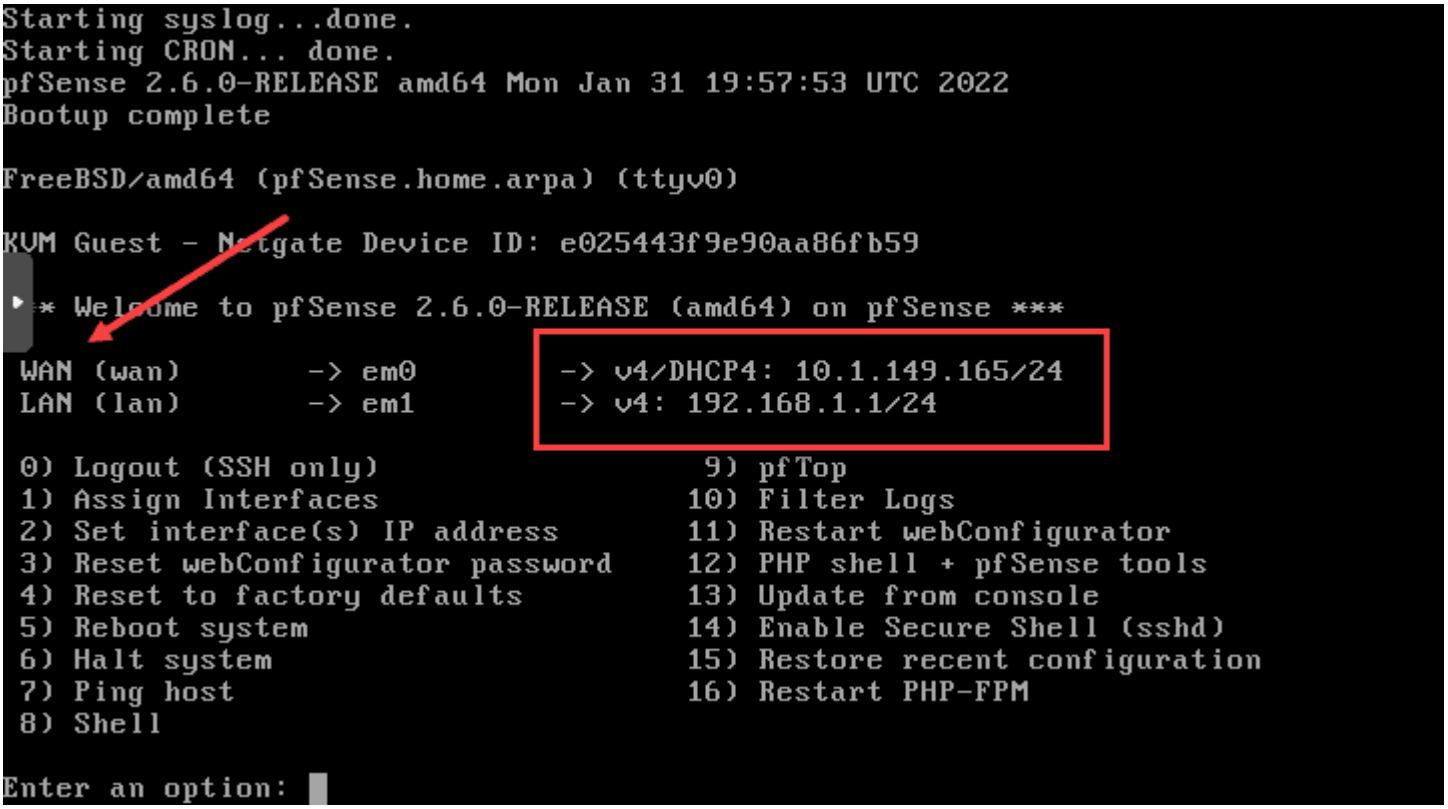
After the pfSense VM boots for the first time, you should see your WAN and LAN interfaces come up and show IP addresses for the WAN and LAN ports. As you can see, these are not on the same network or same subnet.

Most configurations will see the WAN IP address configured from the ISP via DHCP server. You will want to have a static IP address configured on the LAN interface since this will be used as the gateway address for clients connected to the LAN port of the pfSense VM.

The pfSense LAN address is configurable and you will want to configure the address to match your clients. The LAN port also doubles as the management port for pfSense VM by default. You can't manage pfSense from the WAN port by default, only the LAN port. This can be changed later, but is something to note as you run the pfSense virtual machine on your Proxmox box.

The pfSense firewall will also be the default gateway for the clients on the network. The pfSense WAN is the address used for incoming traffic that will be NAT'ed inward to internal IP addresses on the network. For management, specifically note the LAN ip address.

Below, you will note I have private IPs on both the WAN and LAN port. This is because I have this configured in a lab environment. In production, you will have a public IP address configured on the WAN port for true edge firewall capabilities.

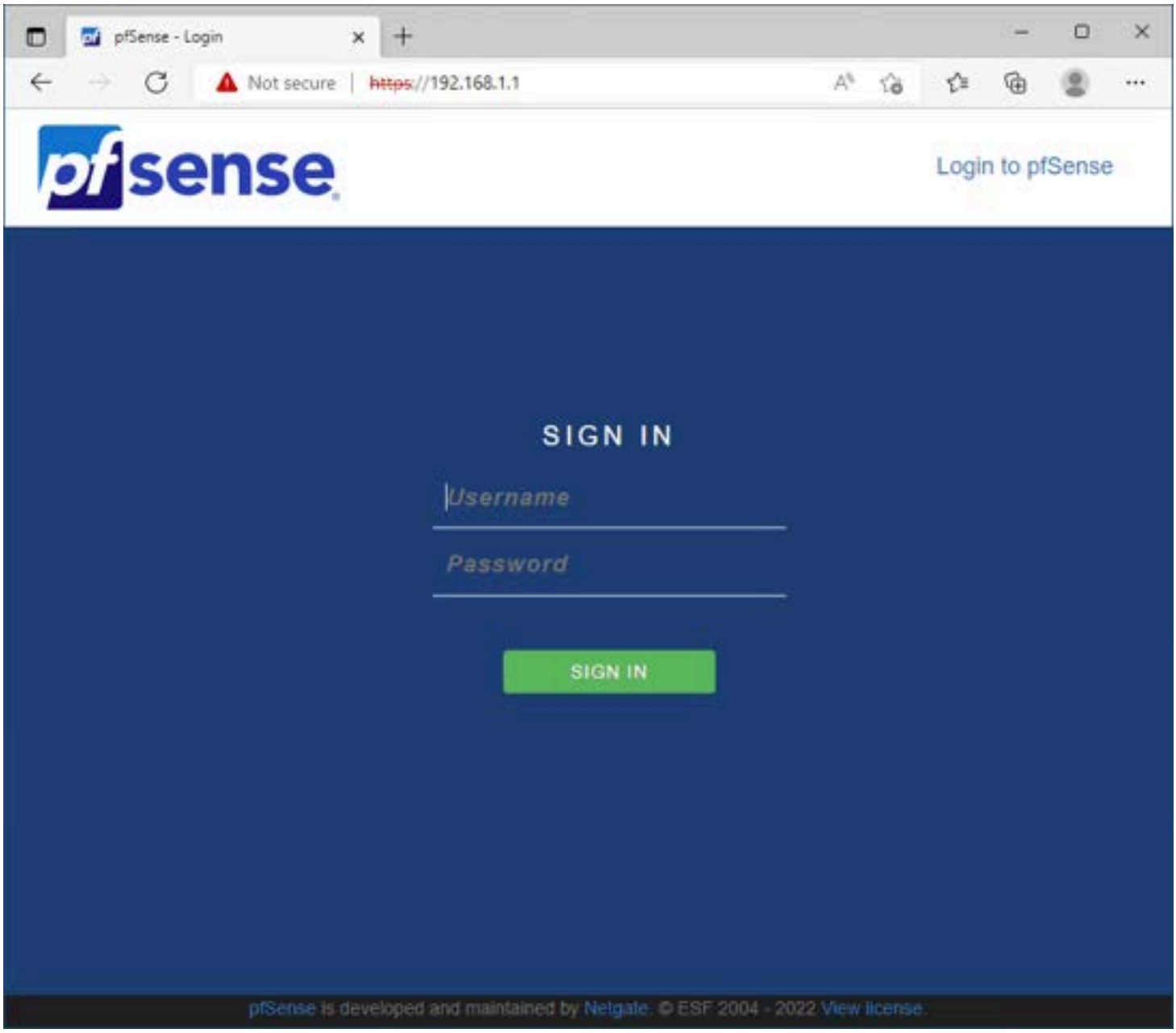


Viewing the interface DHCP address and internal LAN

Configure pfSense VM on Proxmox

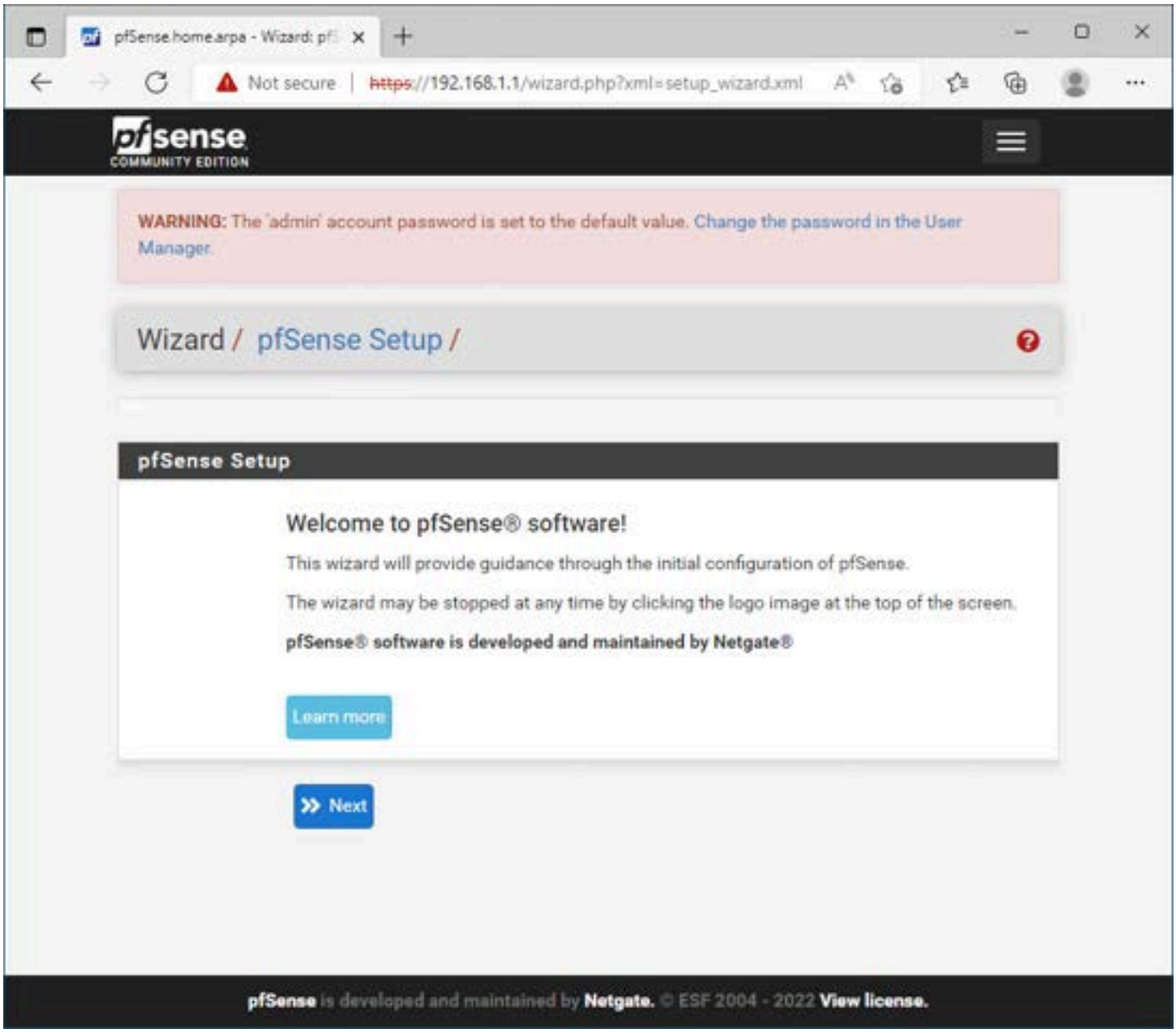
Now, we need to browse out to the pfSense web GUI found on the IP address of the LAN port after installing in Proxmox. The default password will be needed as you log into the pfSense LAN and is:

admin/pfsense



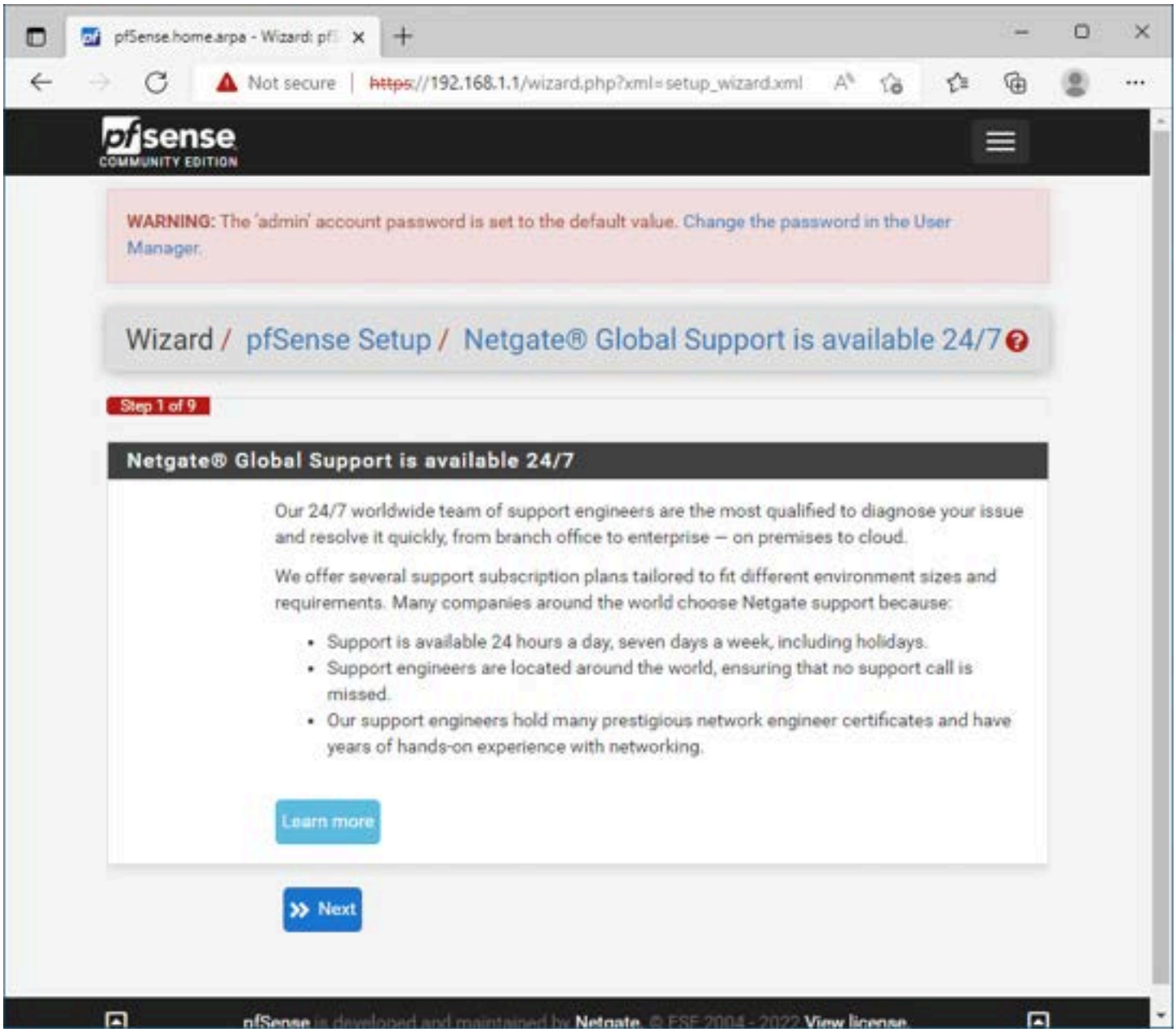
Logging into pfSense VM for the first time

After logging in with the default admin password, the configuration wizard will begin to run pfSense, including the pfSense firewall capabilities.



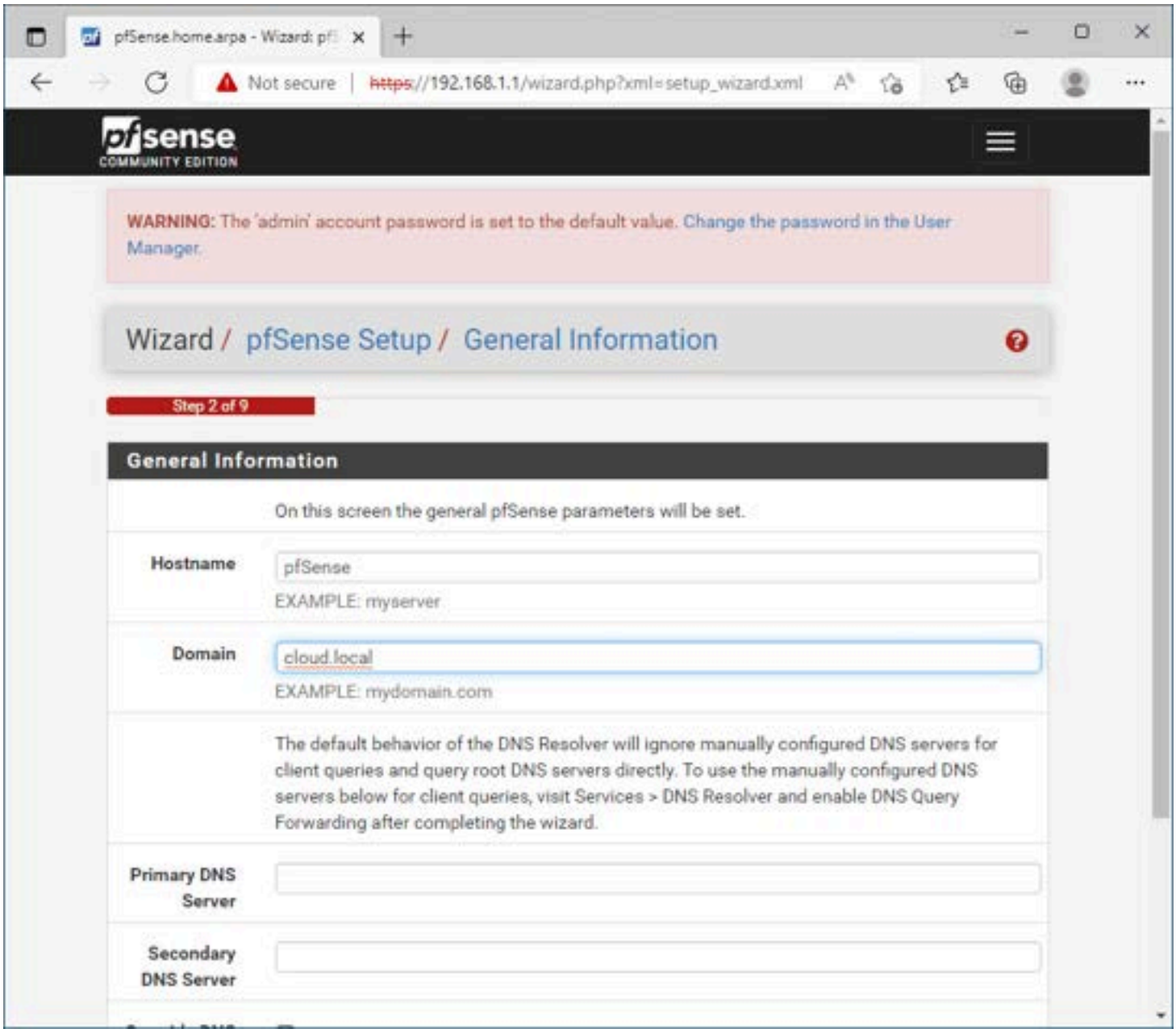
Beginning the pfSense web UI setup wizard

Click next past the Netgate support message.



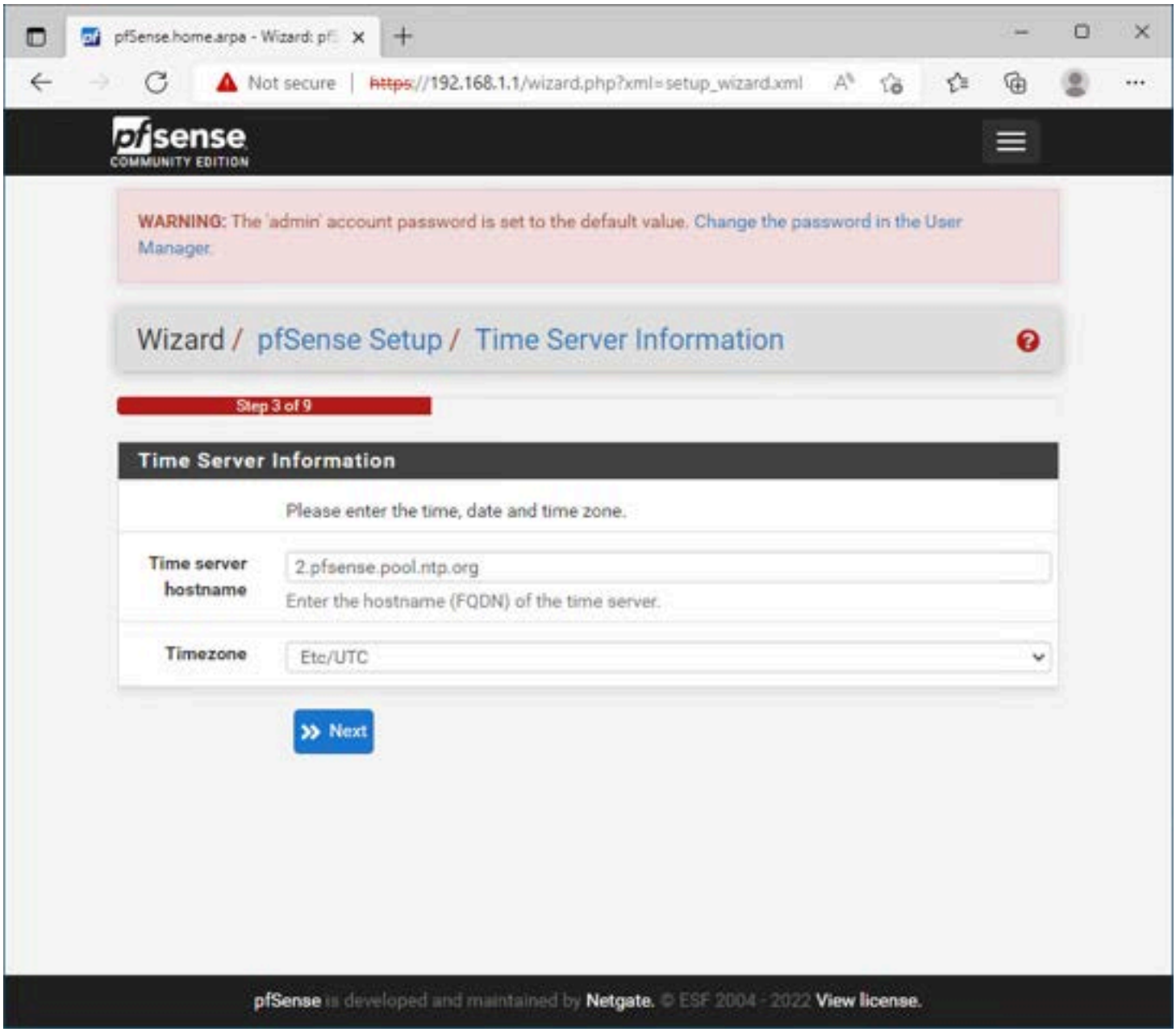
Note the message on Netgate support

Set the pfSense hostname and domain name.



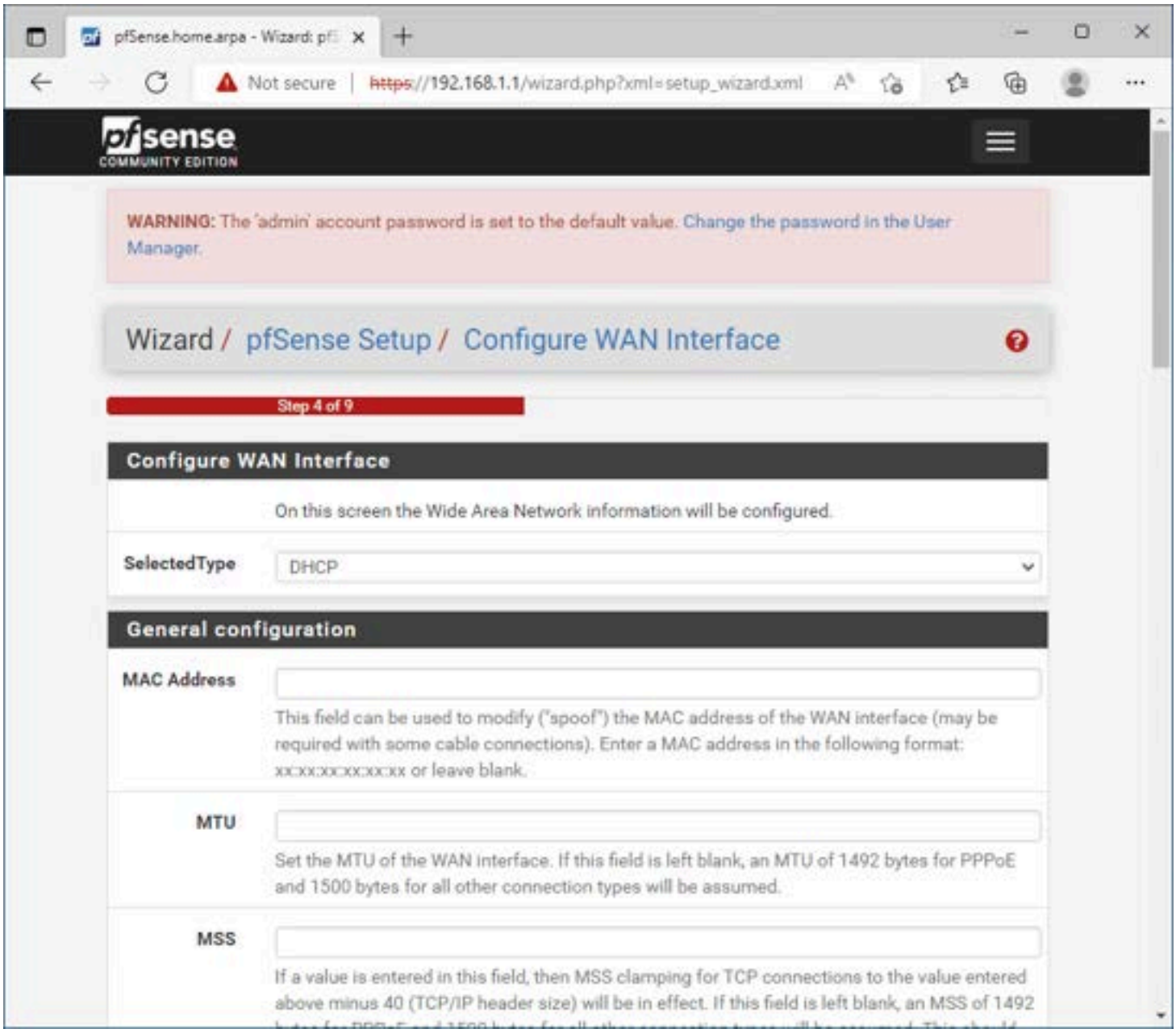
Configure the pfSense hostname

Configure the NTP time server configuration.



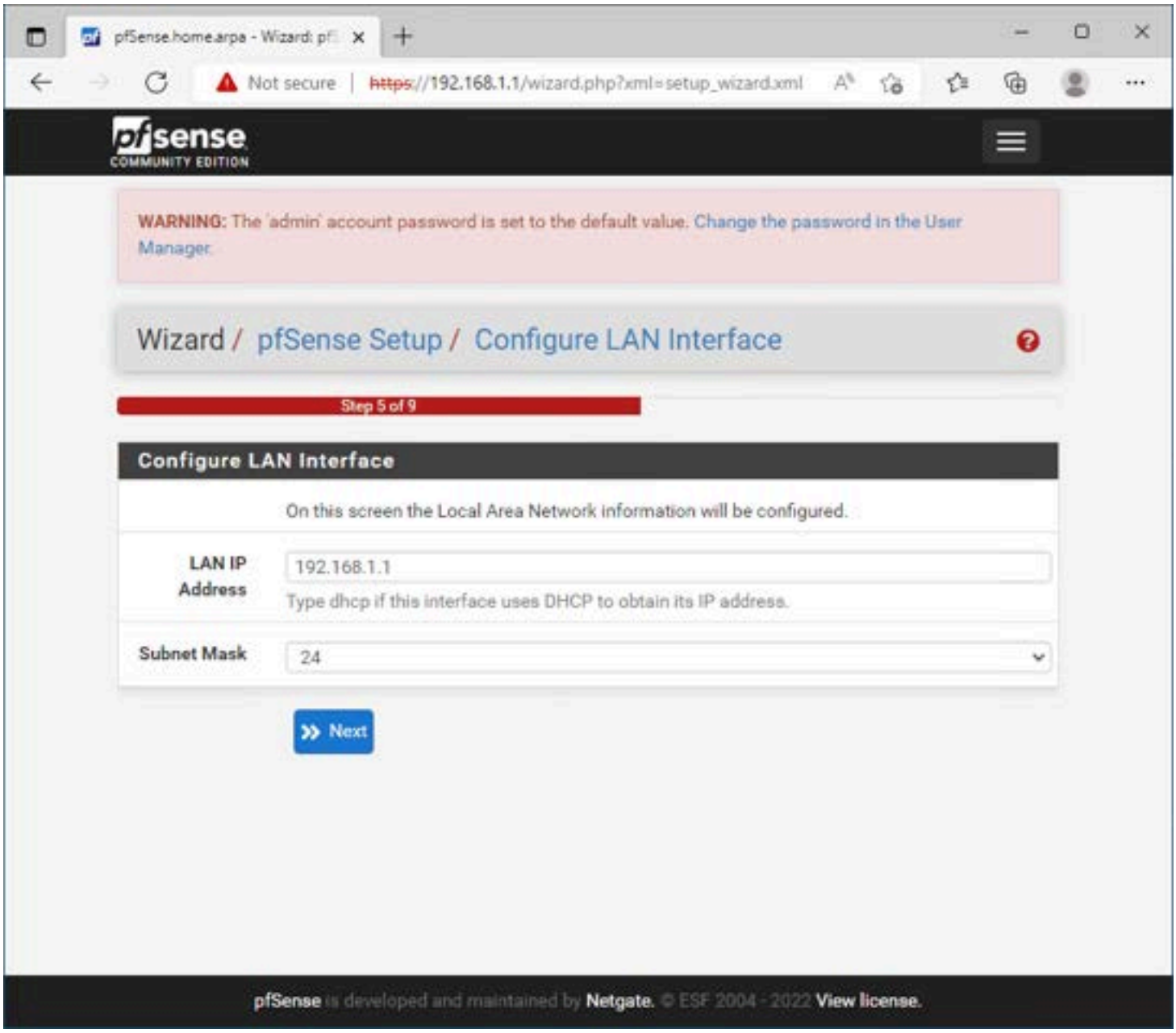
Configure NTP settings in pfSense

Configure the WAN interface. Even though we have already configured this, the pfSense wizard gives you another opportunity to configure the WAN port.



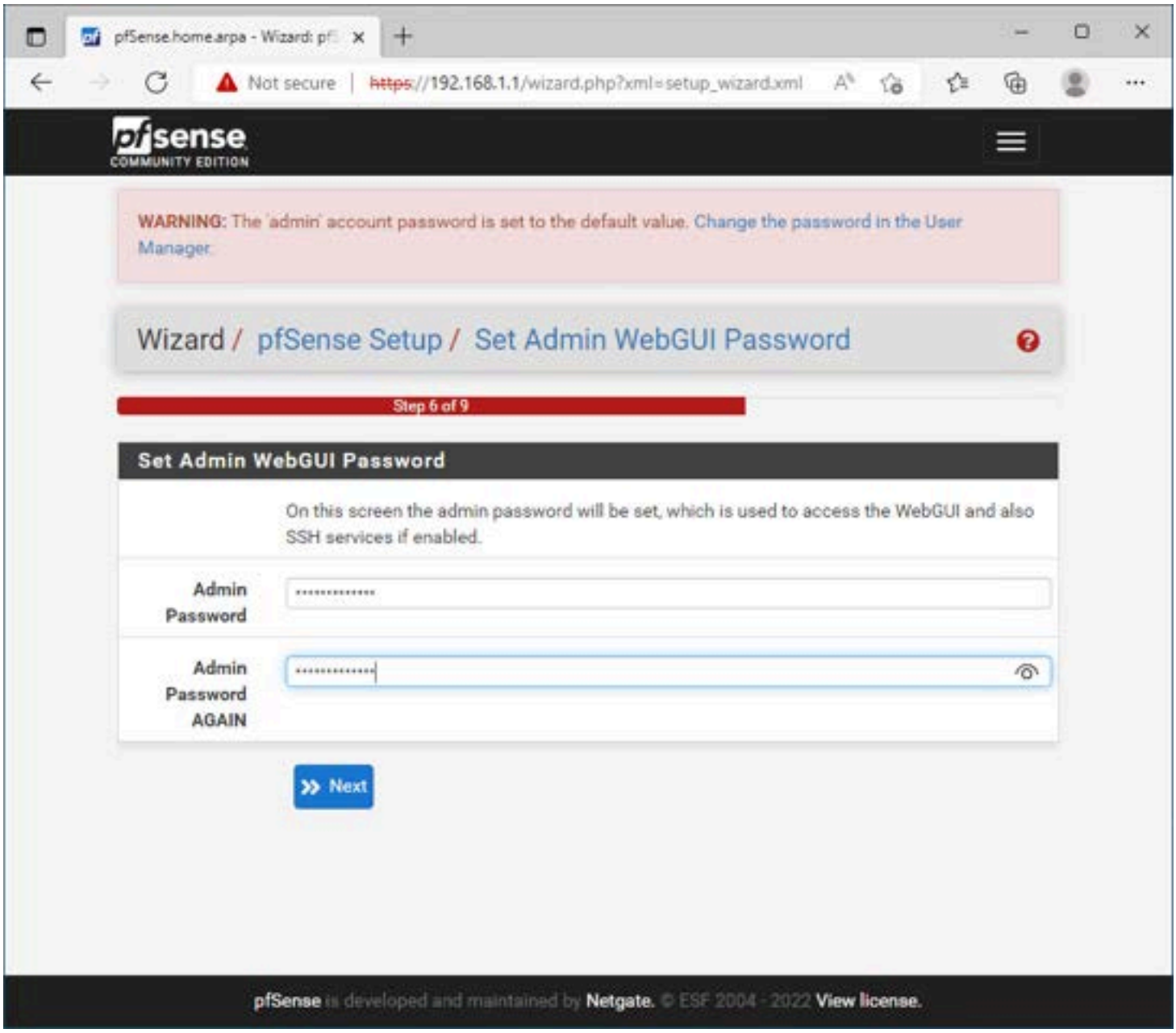
Configure WAN interface in pfSense

Same with the LAN port. You can reconfigure if needed here.



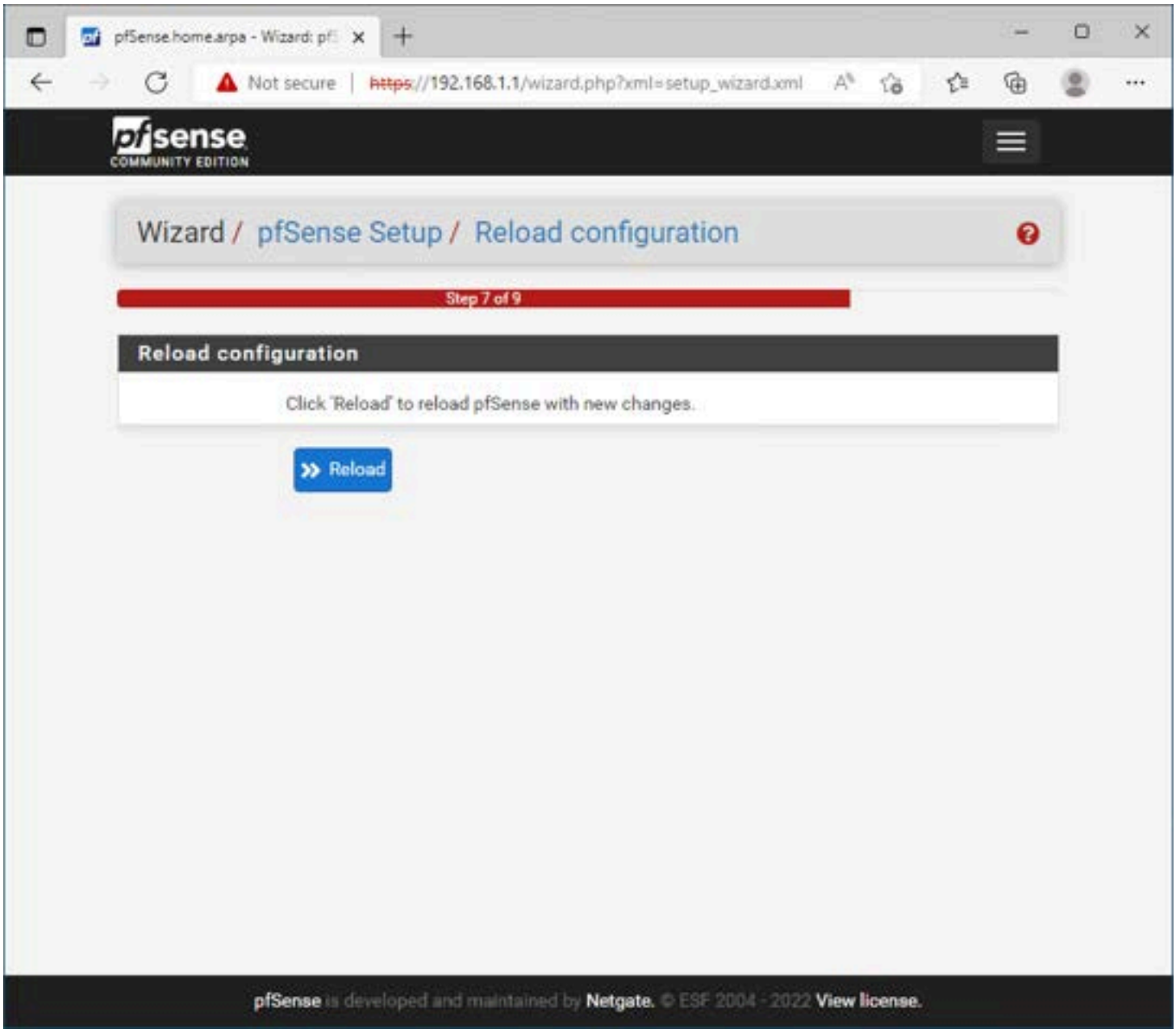
Configure the LAN interface

Change the admin password on the next screen.



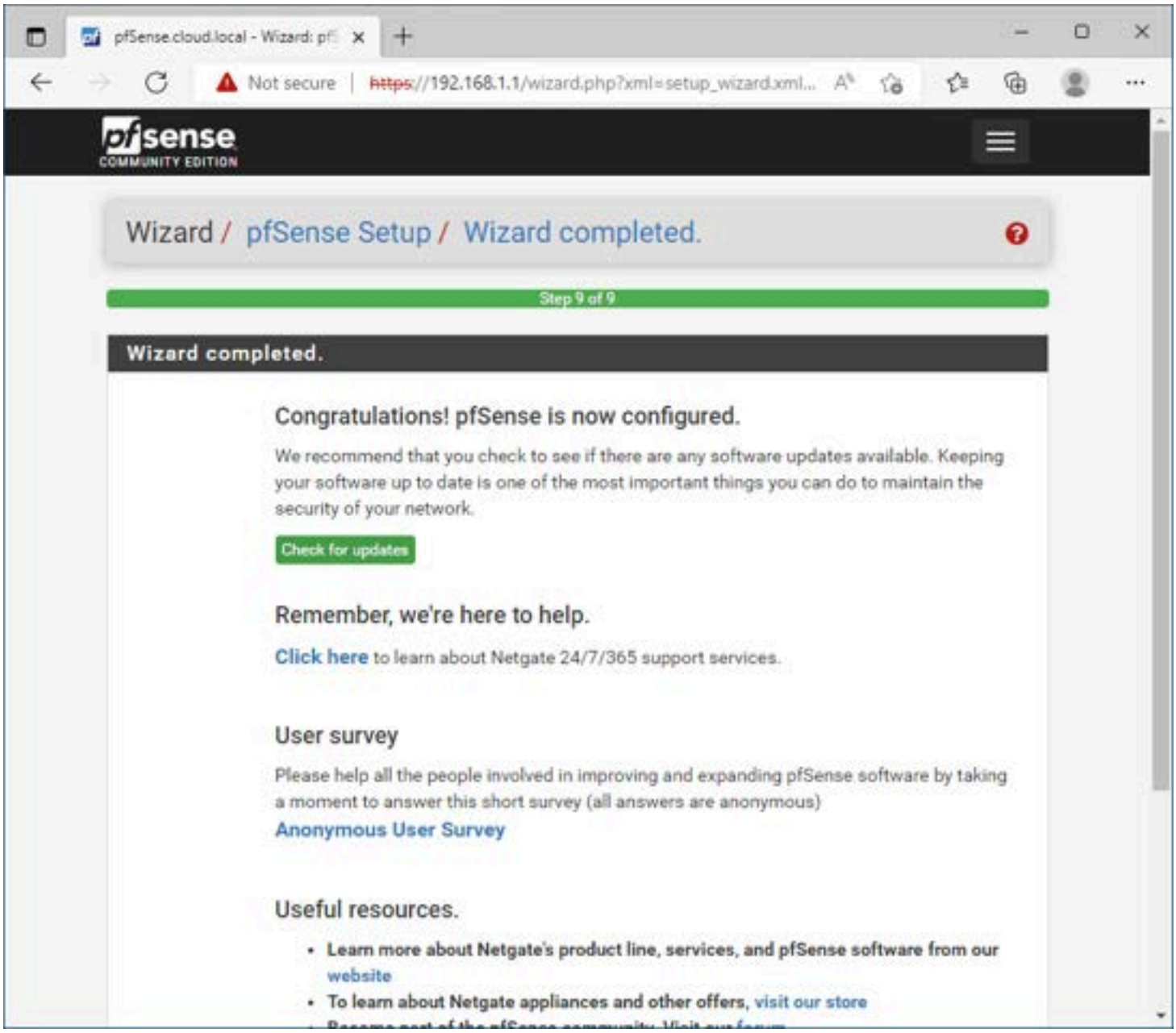
Change the default admin password

Ready to reload pfSense to finalize the configuration.



Reload pfSense with the new configuration

At this point after the reload, the install pfSense process is now complete.



Wizard completes after the reload of pfSense

Congratulations, the install pfSense process is now complete!

Wrapping Up

The pfSense Proxmox installation procedure is straightforward and consists of creating a new Proxmox virtual machine with the correct network adapter settings. Then you power on the VM, run through the initial text configuration setup to install pfSense and establish basic networking connectivity. Afterward, using the pfSense web GUI, you finalize the pfsense installation on Proxmox using the configuration wizard. Proxmox makes for a great platform to install pfSense as Proxmox provides many of the settings and configuration capabilities needed to customize your installation of pfSense Proxmox.

Nested ESXi install in Proxmox: Step-by-Step

December 21, 2023
Proxmox

Vmware esxi on proxmox

If you have a Proxmox VE server in your home lab or production environment and want to play around with VMware ESXi, you can easily do that with Proxmox nested virtualization. Let's look at the steps required for a nested ESXi server install in Proxmox.

Table of contents

- Frequently Asked Questions About Nested ESXi in Proxmox

Nested Virtualization in Proxmox

Nested virtualization in Proxmox VE is easy to set up and has real benefits in learning and setting up rather complex architectures without the physical hardware that would otherwise be needed to set them up physically.

Now, you can use something like VMware Workstation to easily nest ESXi. However, if you already have a dedicated Proxmox host, it is a better platform for a dedicated lab experience. There is always running it on VMware ESXi if you have a physical VMware host.

Proxmox nested virtualization allows exposing the CPU's hardware virtualization characteristics to a nested hypervisor. This process to expose hardware assisted virtualization to the guest ESXi VM is required so the nested hypervisor can run virtual machines.

Preparing your Proxmox VE host to enable nested virtualization for ESXi

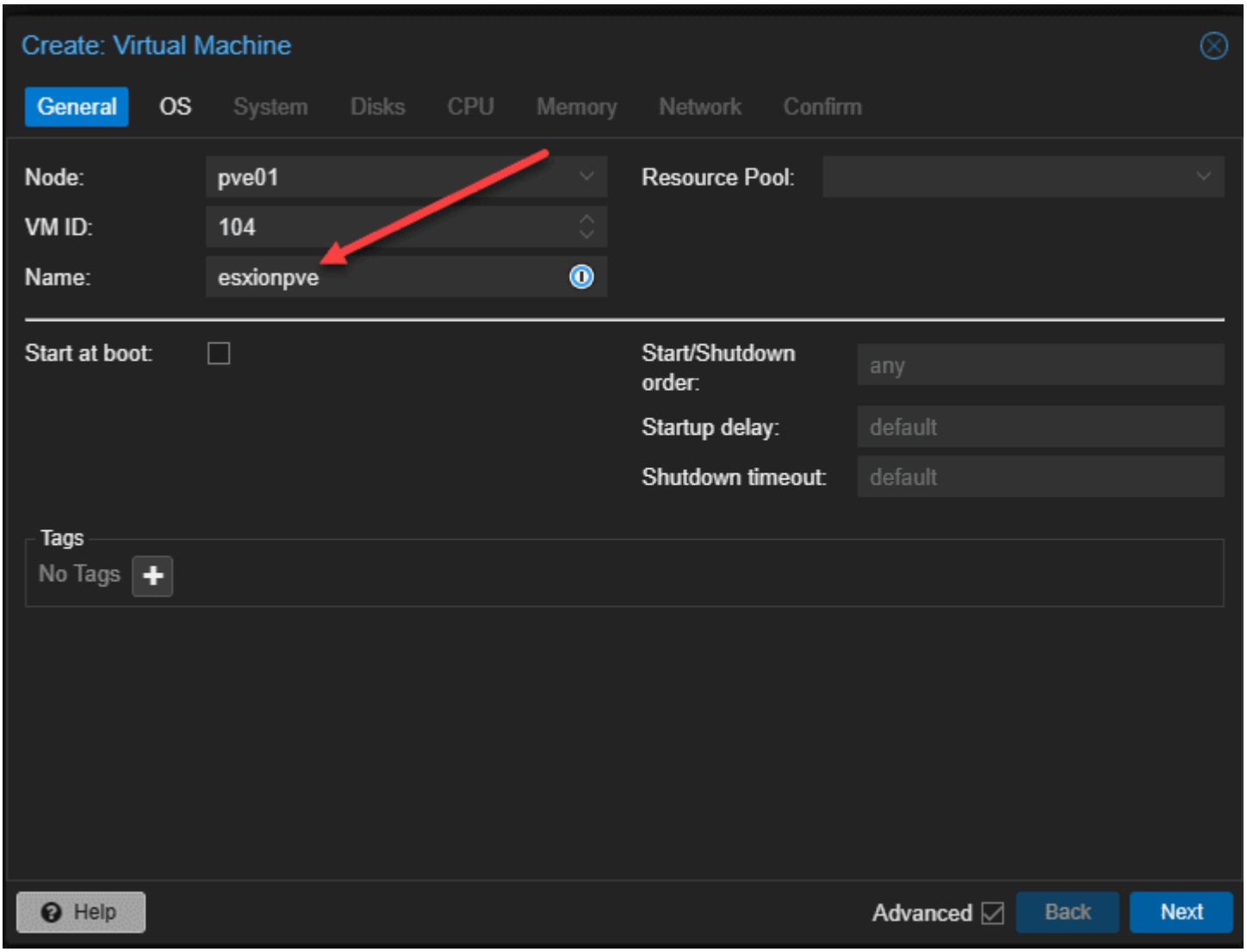
If you don't know how to configure Proxmox Nested Virtualization or enabling hardware assisted virtualization, you can see my recent guide to do that here: How to Enable Proxmox Nested Virtualization.

An overview of the few steps exist to enable nested virtualization for Proxmox and run a nested VM hypervisor are as follows:

- Make sure your CPU supports hardware-assisted virtualization
- Enable hardware-assisted virtualization if it isn't enabled already
- Enable nested virtualization on the nested ESXI installation VM

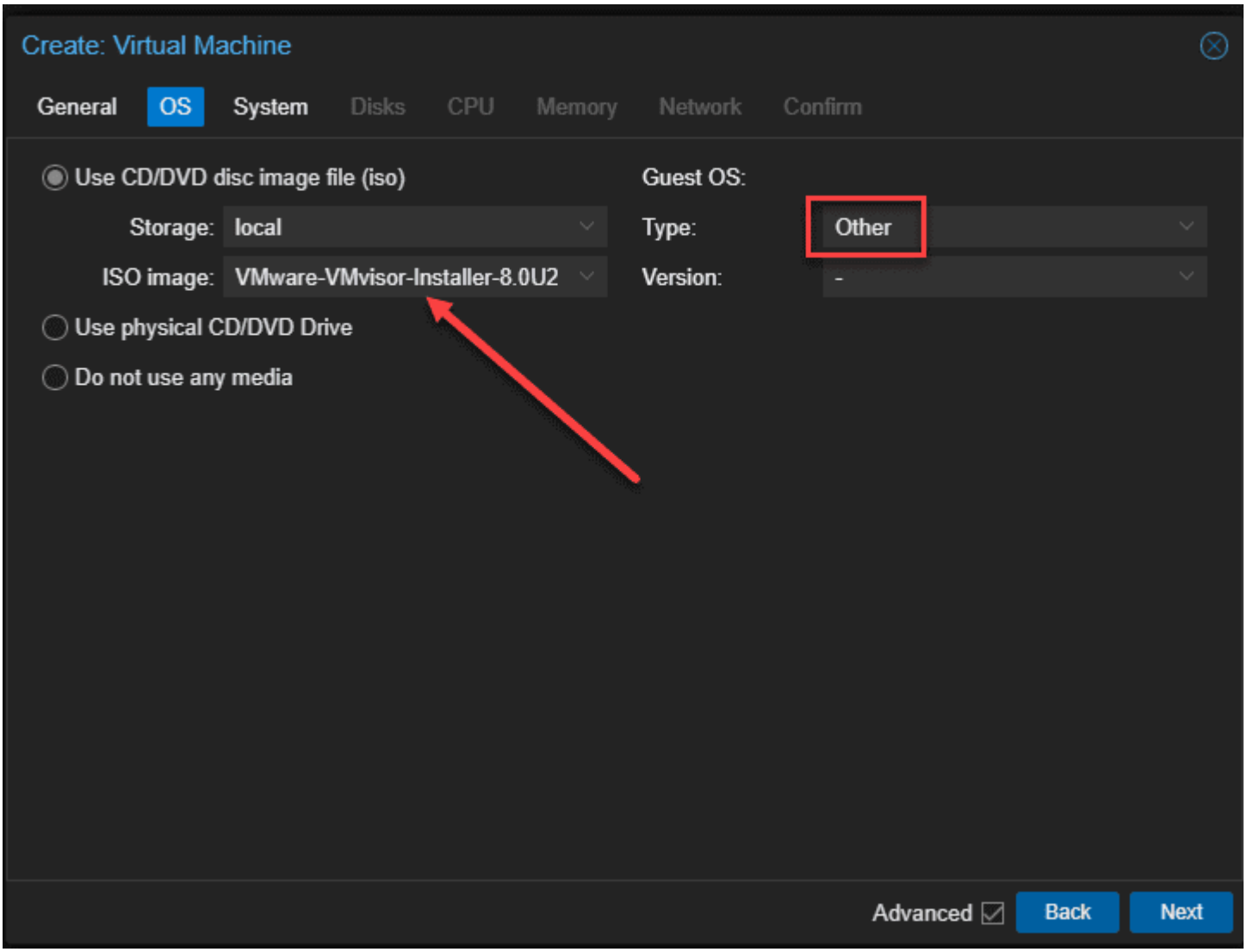
Creating the ESXi VM in Proxmox

VMware hypervisors are extremely popular in the enterprise. Let's look at the process to create the VMware ESXi VM in Proxmox. This is a normal creation process for the most part. I will show you guys one option I chose that didn't work, surprisingly when creating the VM running ESXi.



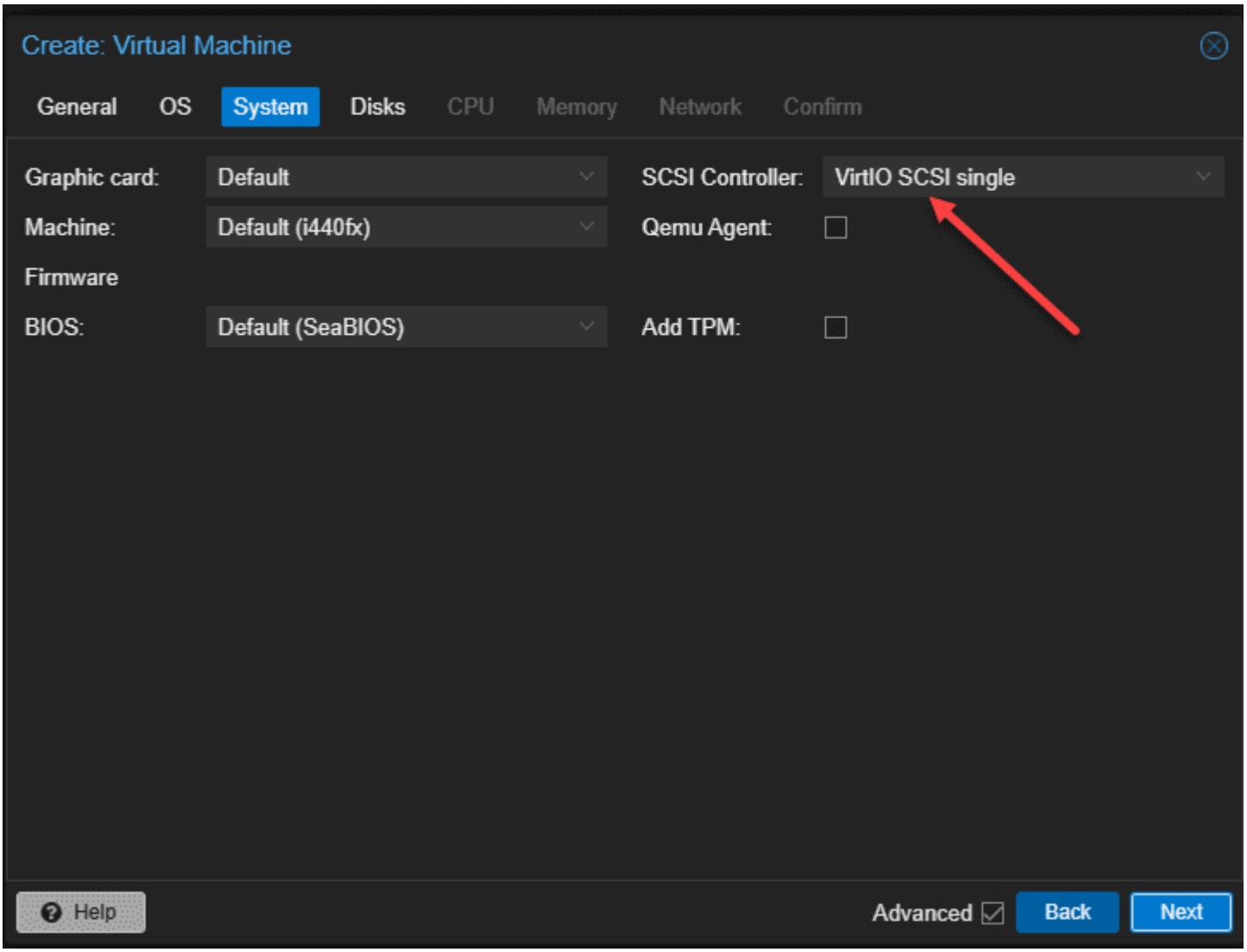
Beginning the create virtual machine wizard

Upload your VMware ESXi 8.0 U2 or other ESXi ISO to your Proxmox server and select this in the wizard. On the type, choose **Other** for the guest operating system.



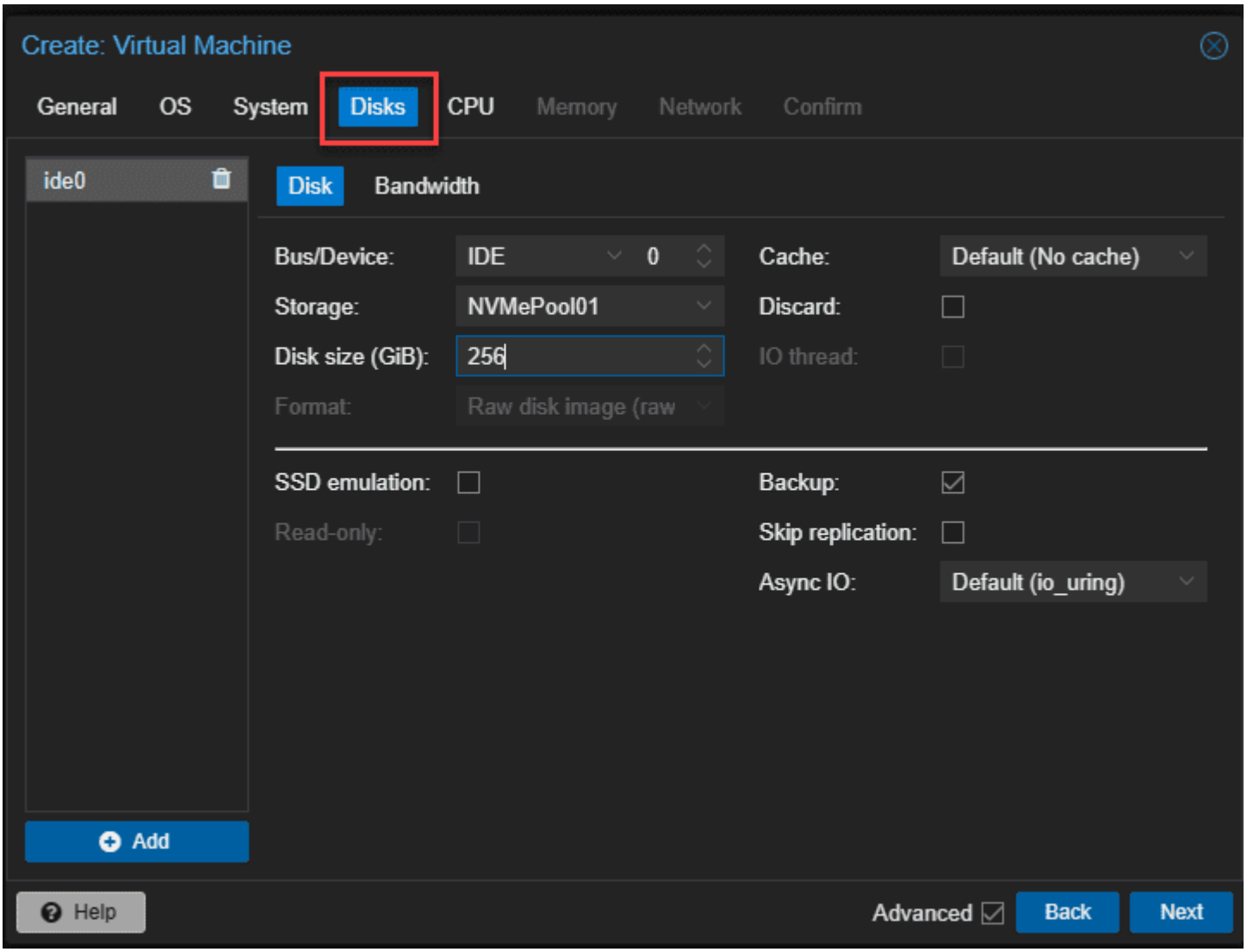
Select your esxi iso image under the os tab

Here I left **VirtIO SCSI single** selected for SCSI controller.



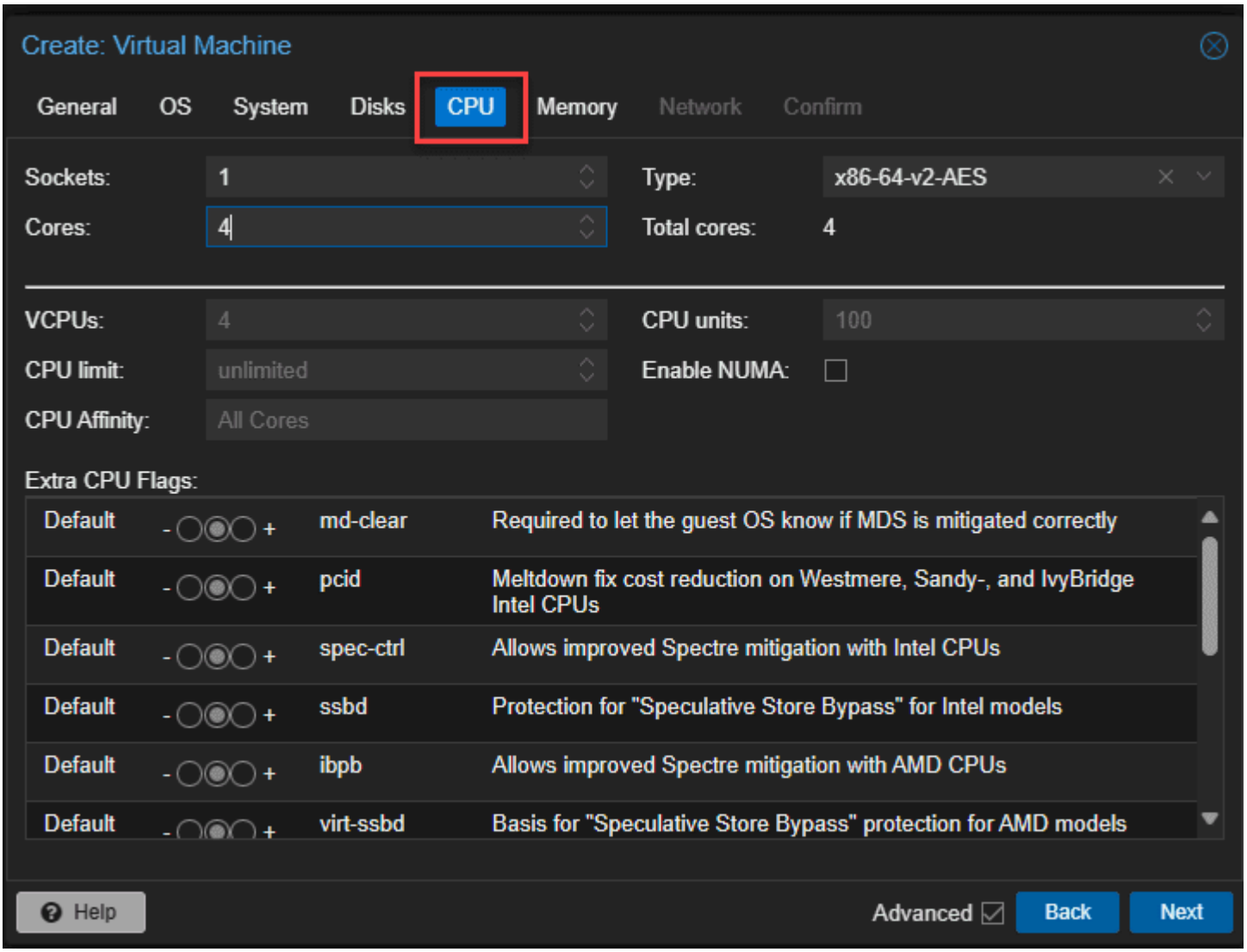
Leaving the defaults under system for esxi nested

On the **Disks** screen, configure the disk size you want and also the Storage location for your VM files and hit **Next**.



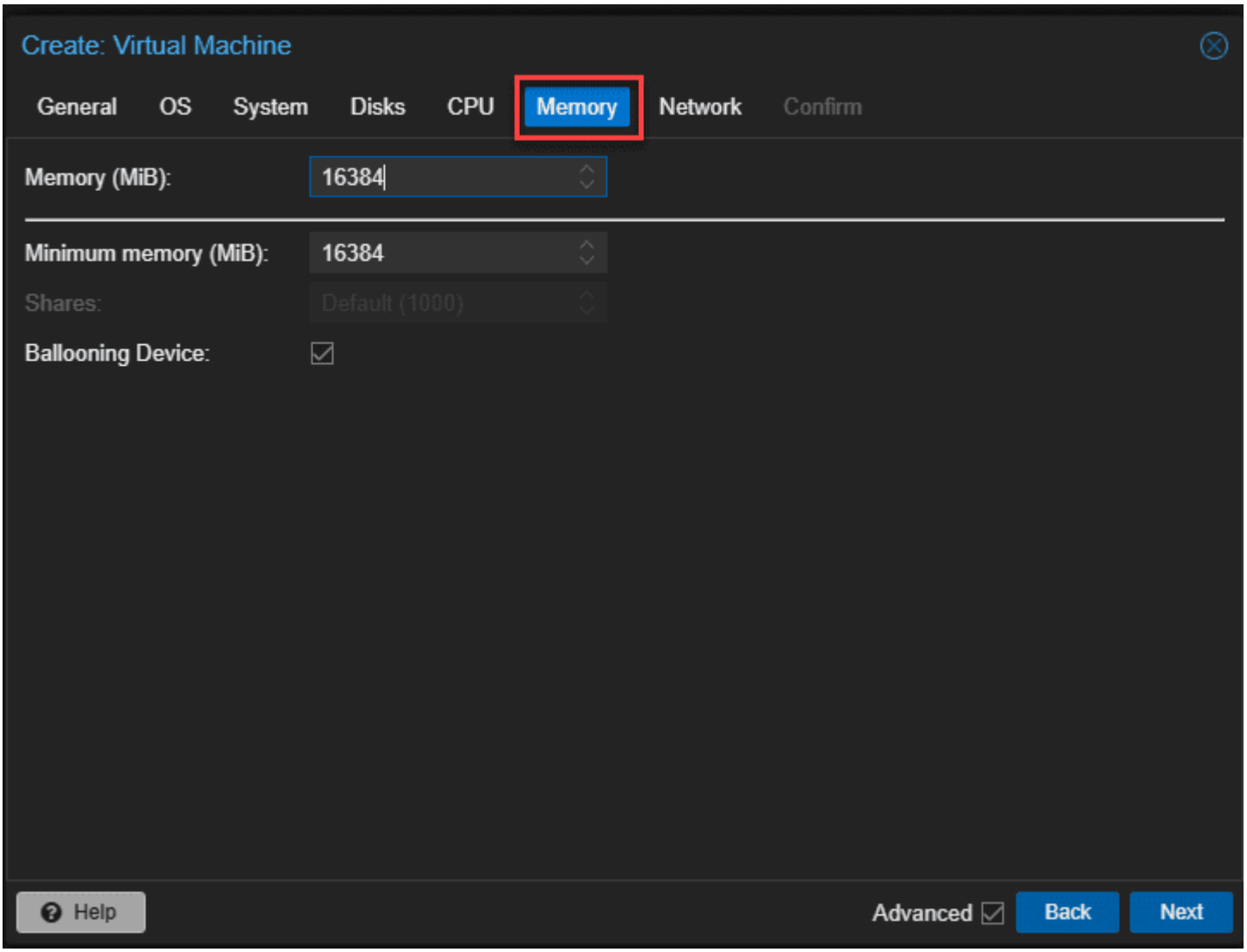
Setting up the storage for the esxi vm

Choose your CPU options.



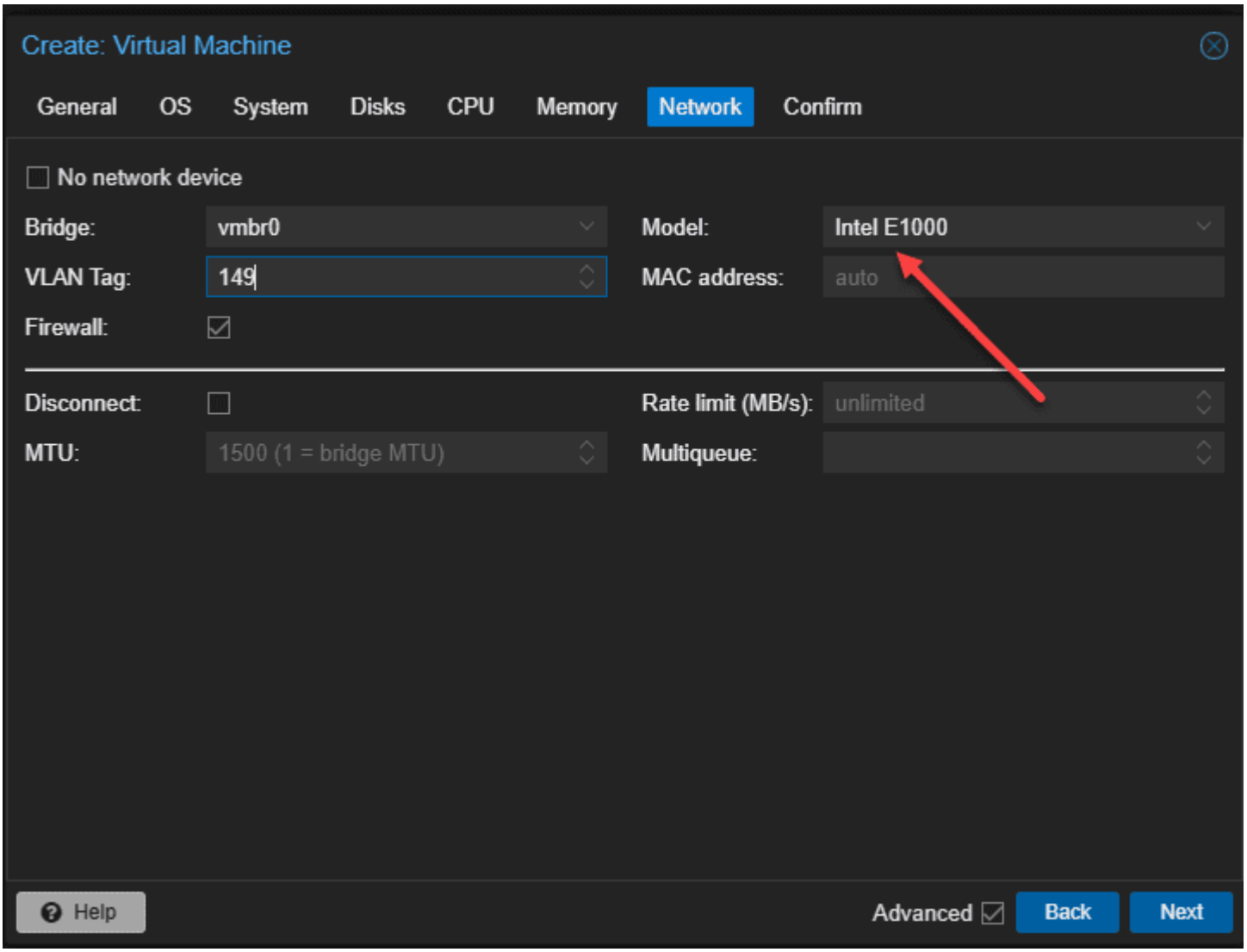
Cpu settings for the esxi nested vm

Configure your memory.



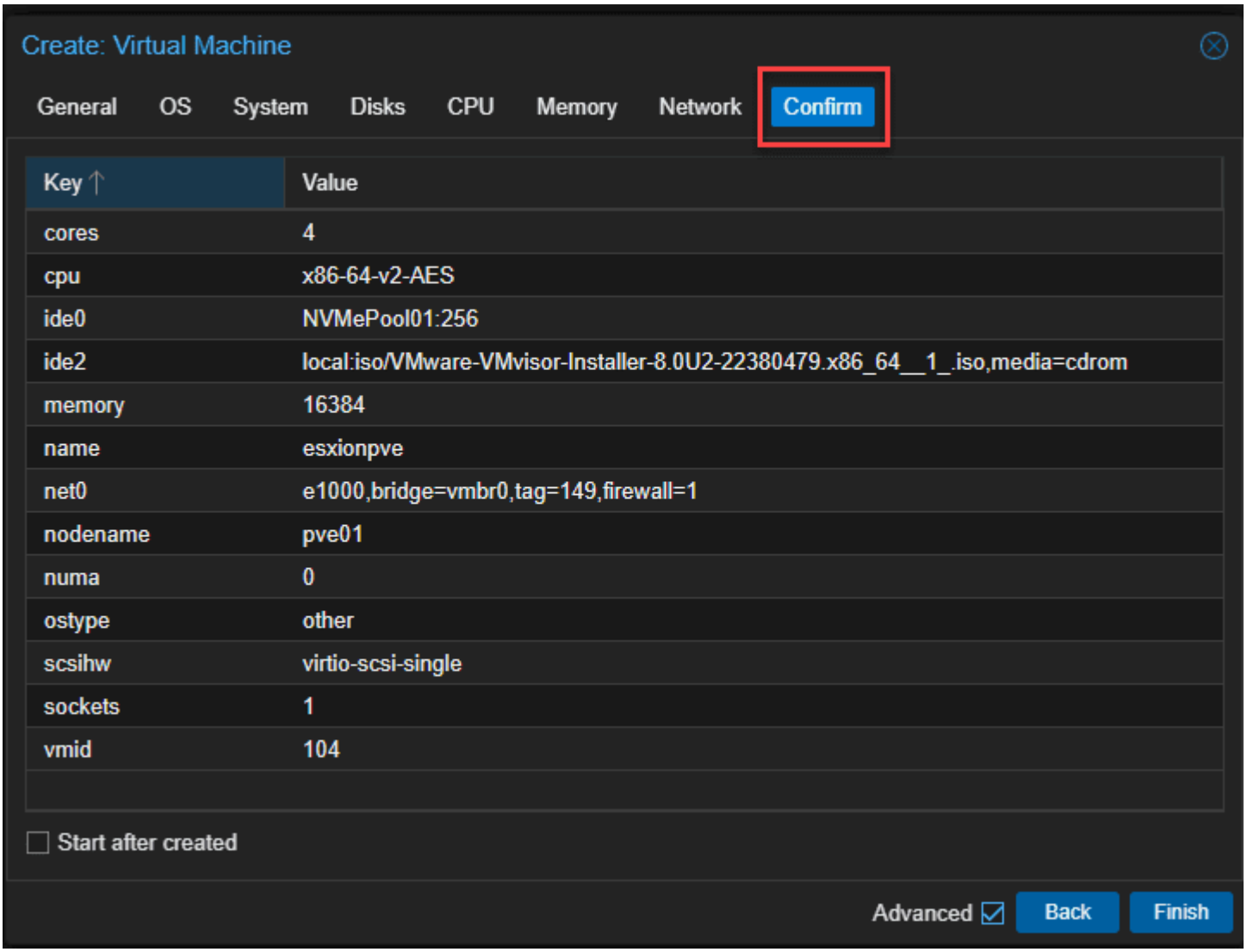
Memory configuration

Ok, so this is the step that surprised me a bit. I here selected **Intel E1000** which is a standard Intel driver. But I will show you what happens during the install.



Setting the network adapter to e1000

Confirm your configuration and click **Finish**.

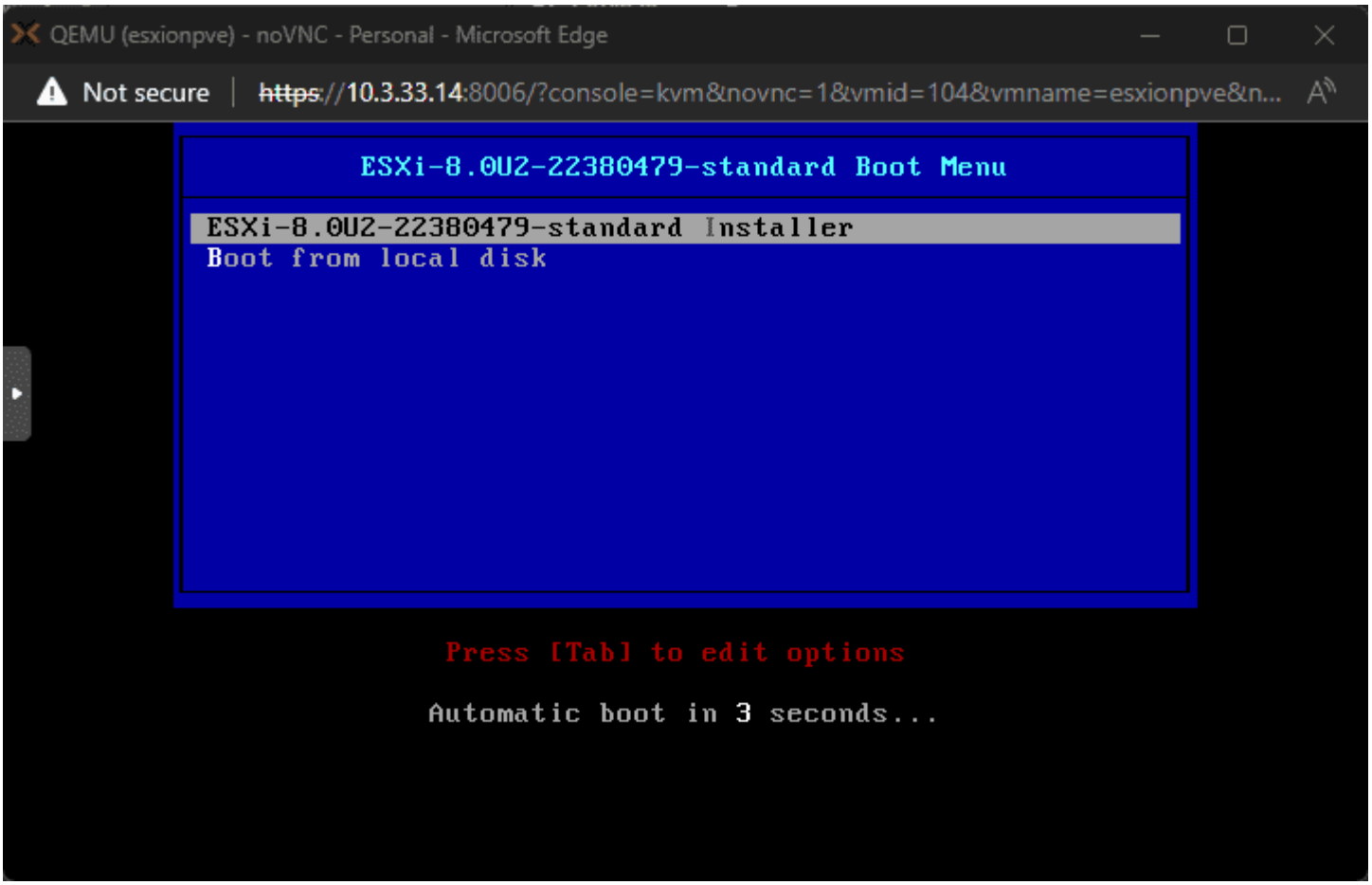


Confirm the installation options

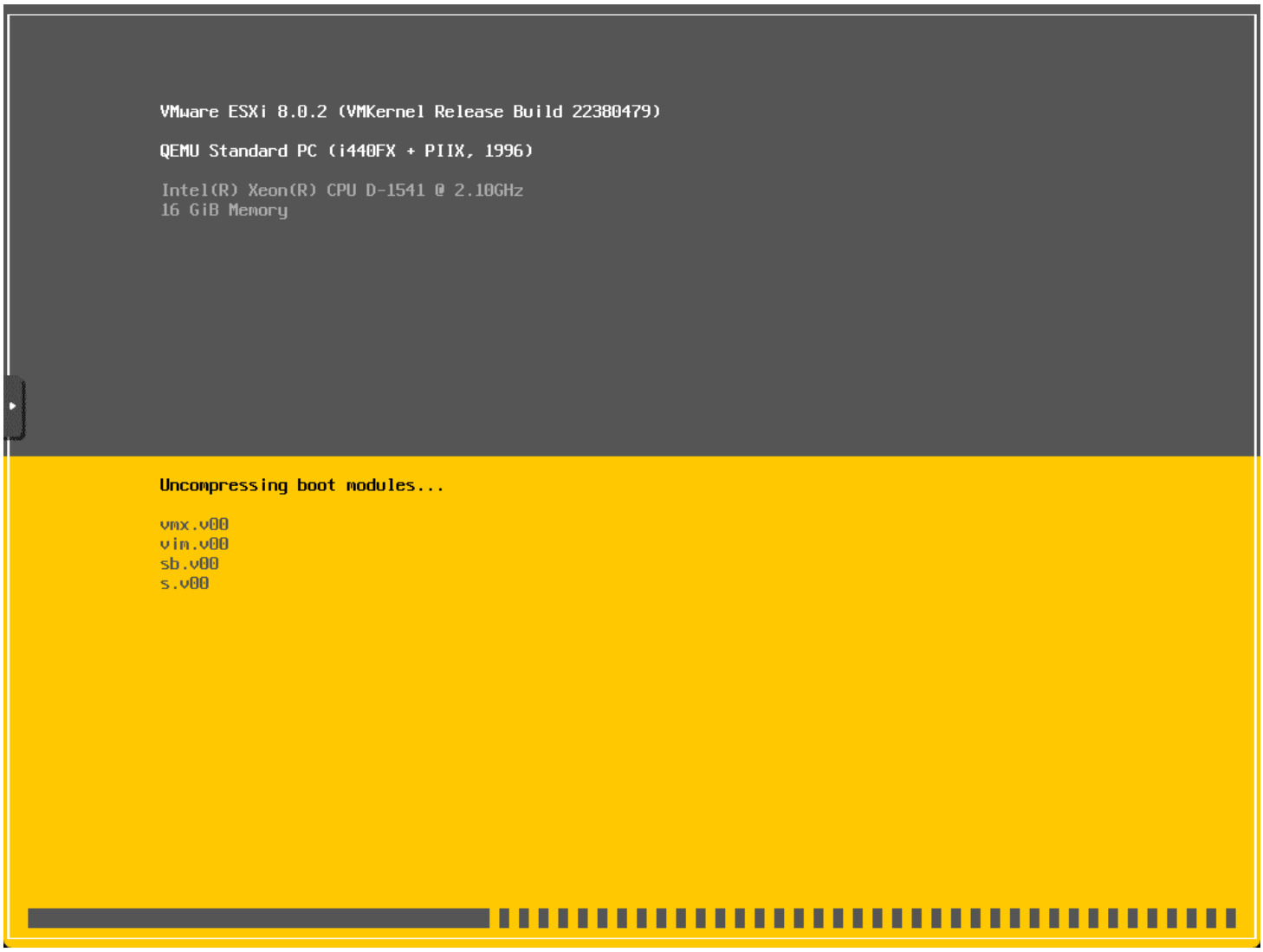
Step-by-Step Installation of Nested ESXi

Let's look at how to install ESXi in Proxmox after we have created the Proxmox virtual machine to house the nested virtual machine install.

Below is booting the VMware VM guest OS in Proxmox.



Beginning the esxi 8.0 u2 installation

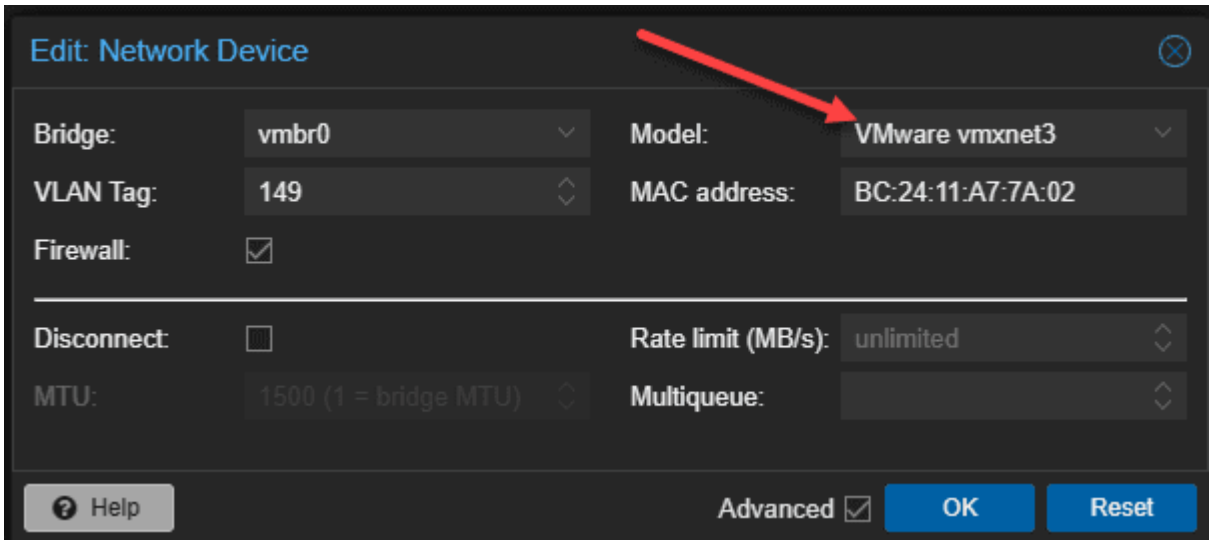


The esxi nested vm boots into the installation

OK, so I told you there was something unexpected happen with the Intel E1000 driver. It didn’t detect the network adapter in ESXi.

No network card detected in esxi

I powered the ESXi VM down and went back and selected **VMware vmxnet3** adapter for the model.



Changing the network adapter model to vmware vmxnet3

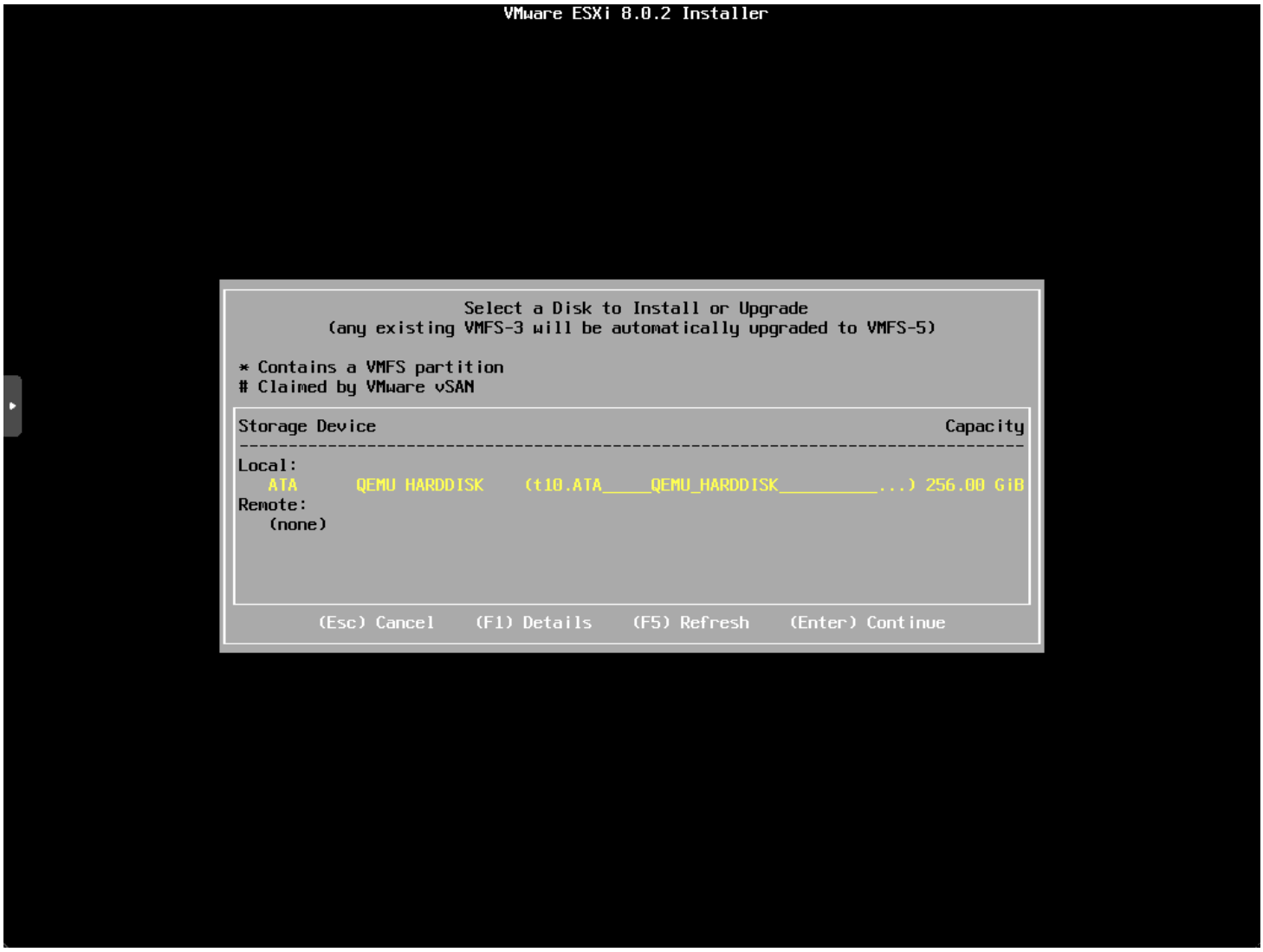
Now, the network adapter was recognized and the installation proceeded.

The installation of nested esxi continues

Now for the standard screens, but we will show them anyway. Accept the EULA.

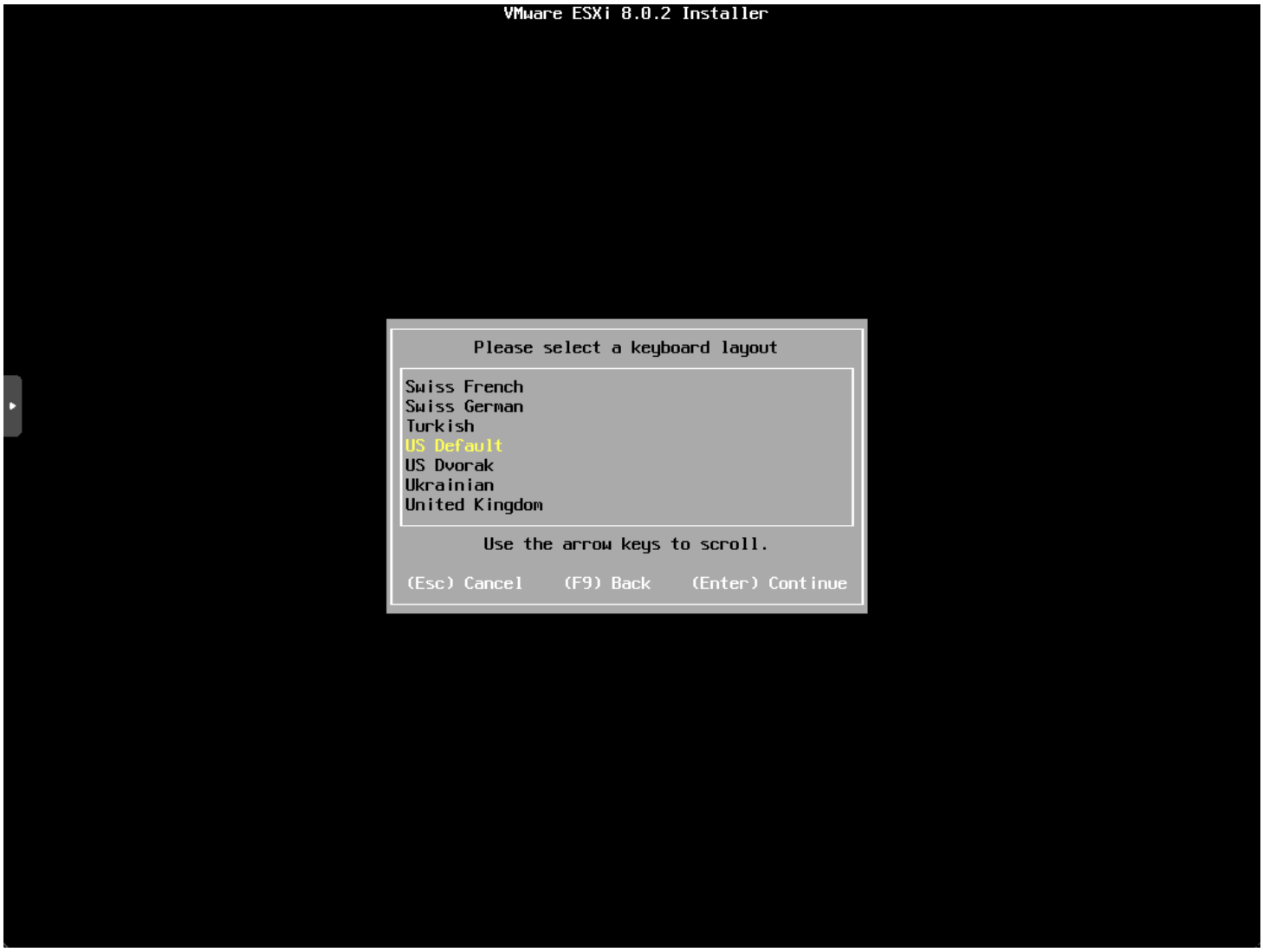
Accept the eula 1

Select the target storage for the installation.



Select the installation target storage for nested esxi

Select the location for the keyboard layout.



Select the esxi keyboard layout

Enter and confirm your password.

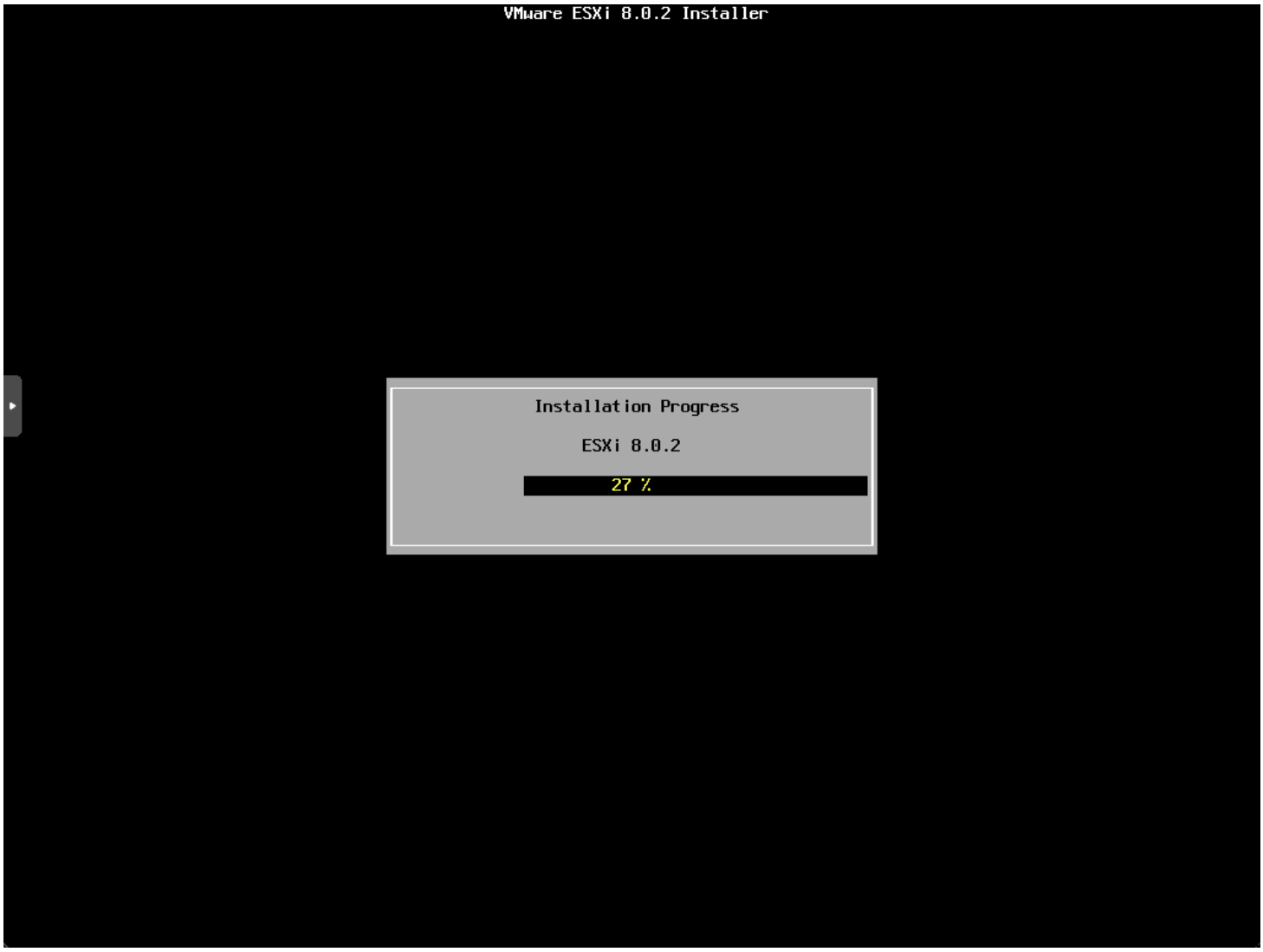
Configure the root password for esxi

I am running on an older Xeon D processor so we see the alert about an outdated processor that may not be supported in future releases. You will see the same error on bare metal.

Warning about older cpu support in esxi 8.0 update 2

Confirm the installation of esxi and repartitioning

The installation begins.

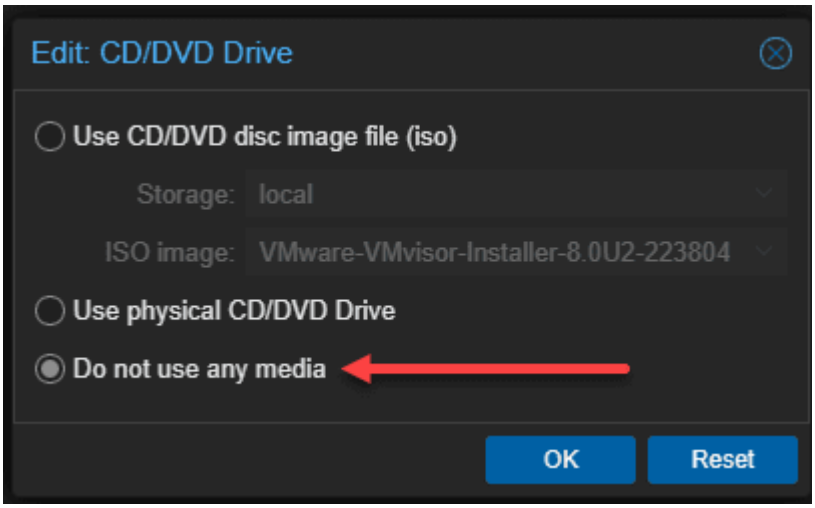


Esxi installation progress begins

Finally, we are prompted to remove the installation media and reboot.

Installation finished remove installation media

Hopping back over to Proxmox, I remove the ESXi ISO before rebooting.

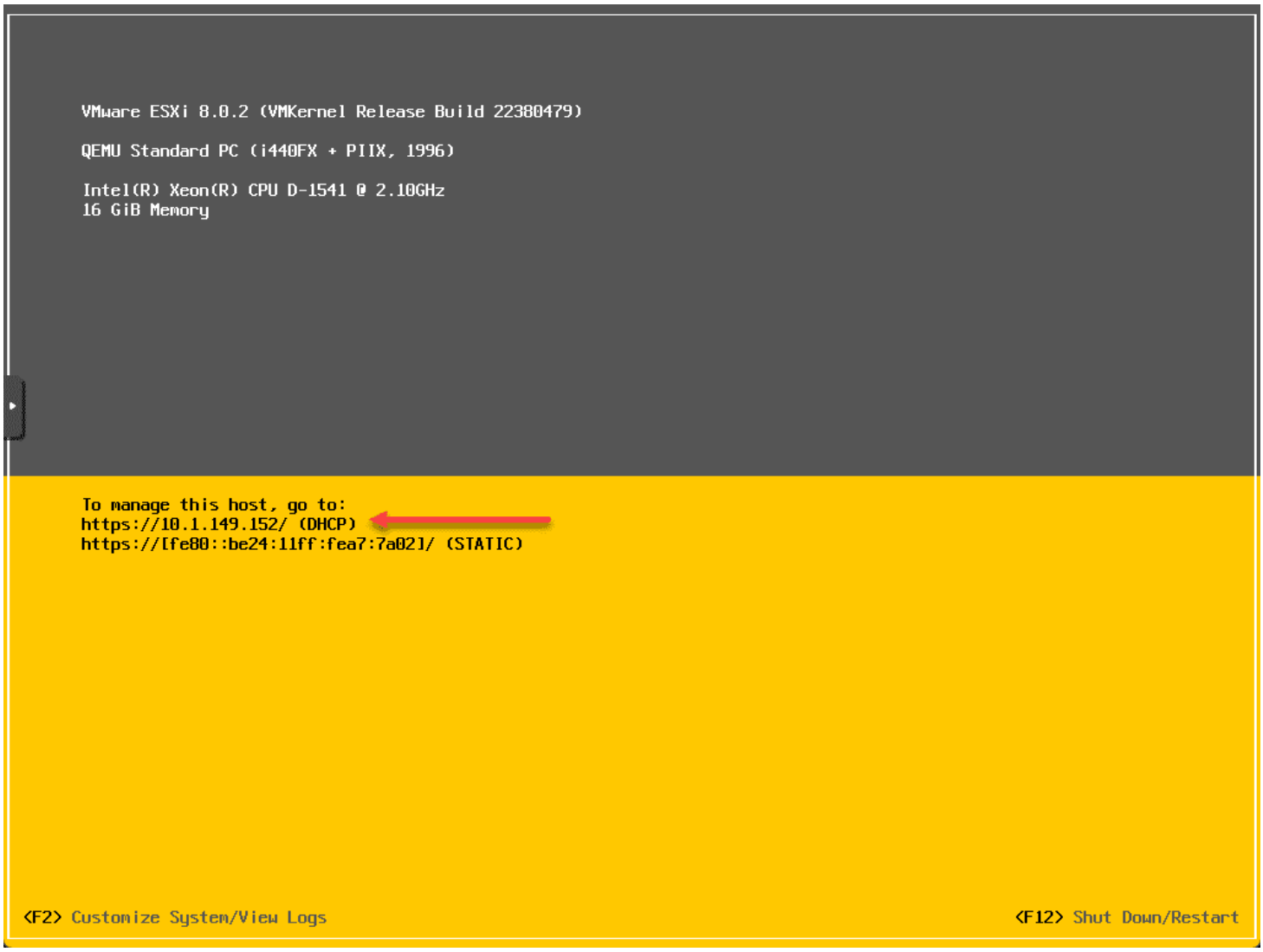


Removing the iso from the esxi vm in proxmox

After initiating a reboot.

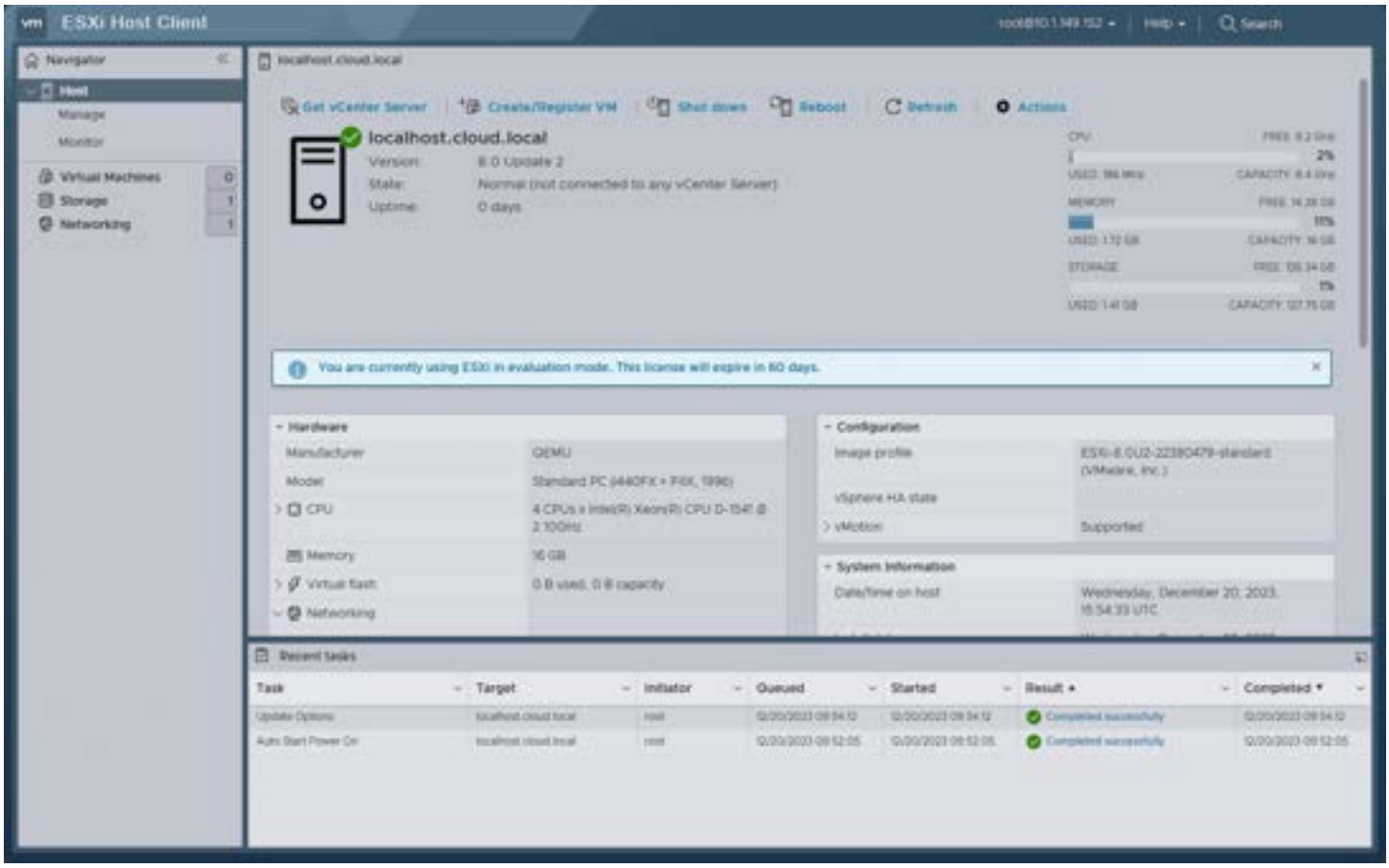
Rebooting the esxi installation

After the nested ESXi installation boots, we see it has correctly pulled an IP address from DHCP so the network adapter is working as expected.



Vmware esxi vm in proxmox boots and it correctly pulls a dhcp address

Below, I logged into the VMware host client to manage the ESXi host running in Proxmox.



Logged into the esxi host client

Managing Virtual Machines in a Nested Setup

The cool thing about working with ESXi that is nested in a Proxmox VM is that, for the most part, you won't notice much difference if you are used to accessing the ESXi host client or adding the ESXi host to the vCenter Server and managing it with vCenter.

Using advanced features in nested VMs

The great thing about running ESXi as a nested hypervisor, is you won't see any difference in the advanced features for nested VMs. You will still be able to do things like installing VMware Tools in Linux and your Windows Server operating system instances.

If you are configuring a cluster of ESXi hosts with vCenter, you can utilize features like vMotion and DRS within a nested VMware vSphere cluster.

Troubleshooting Common Issues in Nested Environments

Running nested ESXi in Proxmox can be a bit of a mind-bender on the networking side. However, this is not unique to Proxmox, as running nested ESXi on a physical ESXi host can be the same challenge.

First, though, you need to understand Proxmox VLANs. I just covered this recently as well. So, check out my post on Proxmox VLANs to first understand how to configure VLANs in Proxmox.

Just remember, on the nested VMware ESXi side, you can't tag VLANs on your port groups as this will lead to "double tagging". They will instead assume the tag from the Proxmox side.

What I like to do is set up the Proxmox Linux Bridge as a trunk bridge, which is the default configuration when you make it VLAN aware. Then, you can change the tag on your network adapter configured for your VMware ESXi VM to tag the traffic from the ESXi VM.

Frequently Asked Questions About Nested ESXi in Proxmox

How Does Nested ESXi Differ from Regular Virtualization in Proxmox?

Nested ESXi in Proxmox takes virtualization a step further by running a virtual machine (VM) within another VM. In nested setups, ESXi acts as a guest hypervisor within the VM to create and manage additional VMs in this second layer of virtualization.

Can I Run VMware Tools in a Nested ESXi VM?

Yes, VMware Tools can be installed and run within a VM running on nested ESXi in a Proxmox environment. This installation enhances the functionality and performance of the nested VMs. It provides better hardware compatibility and improved management capabilities.

What Are the Key Considerations for VM Hardware Settings in Nested Virtualization?

When configuring VM hardware in a nested virtualization setup, it's important to allocate sufficient resources, such as CPU and memory, to ensure smooth operation. Additionally, you should enable promiscuous mode in the virtual switch settings to allow communication between nested VMs.

Is Nested ESXi Suitable for Production Environments?

Not really in most scenarios. You definitely won't be supported by VMware in a nested environment and likely not Proxmox either. It is best to keep nested environments in their proper place, for learning and labbing and testing out configurations without the physical hardware to install on bare metal.

How Can I Optimize the Performance of Nested VMs in Proxmox?

Give attention to resource allocation, enabling hardware-assisted virtualization, and configuring network settings properly. Monitor your Proxmox VE host and nested ESXi VMs to make sure there are no performance issues.

Can Windows Server Be Used Effectively in a Nested ESXi Setup?

Windows Server can be run as a guest operating system in a nested ESXi VM. This setup allows for testing and development of Windows-based applications in a controlled, virtualized environment, leveraging the capabilities of both Proxmox and ESXi.

Are There Specific Network Configurations Required for Nested ESXi in Proxmox?

Nested ESXi in Proxmox requires specific network configurations, including setting up virtual switches and enabling promiscuous mode to allow proper network traffic flow between nested VMs. Proper configuration ensures seamless connectivity and communication within the nested environment.

What Are the Benefits of Using Intel VT-x in Nested Virtualization?

Using Intel VT-x in nested virtualization enhances the performance of nested virtualization. This technology enables more efficient emulation of hardware features. Really, you don't want to use nested virtualization without it.

Wrapping up

Hopefully, this blog post has been a help to any who are running Proxmox as your hypervisor running your home lab environments. It is easy to get a virtual machine running with VMware ESXi in a Proxmox nested environment. Keep in mind the need to use the VMware vmxnet3 adapter and the note on Proxmox VLAN tagging. If you are running guest VMs in your ESXi VM, you will also need to keep in mind the need to enable promiscuous mode for your Proxmox bridge.