CLOUD COMPUTING FROM THE GROUND UP WITH PROXMOX 9

Making Proxmox Accessible and Understandable

Document Version: 2_02

Installing, Configuring and Operating Proxmox

Contents

Introduction

I started creating this collection as an internal "note to self" while I was learning PVE. Later it became apparent that many users, with varying degrees of experience, also needed VMware alternatives.

One of the most obvious aspects of learning PVE is that almost all the resources focus on CLI execution of everything, from the forums to the wiki and especially the documentation. While not adverse to the CLI myself, I also know that many users struggle.

Because of this, I set about to convert my "note to self" to an extensible training manual that could be applied in my own lab environment or easily journaled into any other lab environment. I have based the SBS LABs in this book on my own experience in establishing many PVE environments and assisting Organizations in migration from other platforms, primarily VMware.

My primary goal is the dissemination of information in a real-world, easy-to-use format. As such, I will be using the GUI wherever possible but leveraging the CLI where needed. In the process, providing examples and screenshots of each step, along with the results.

I also will be publishing this work using the Creative Commons license. This license requires that re-users give credit to the creator. It allows re-users to distribute, remix, adapt, and build upon the material in any medium or format, even for commercial purposes. If others remix, adapt, or build upon the material, they must license the modified material under identical terms.

I welcome all input and will provide credit/acknowledgement for anything I use. Please visit: https://pvemanual.com/forum/ to connect and contribute.

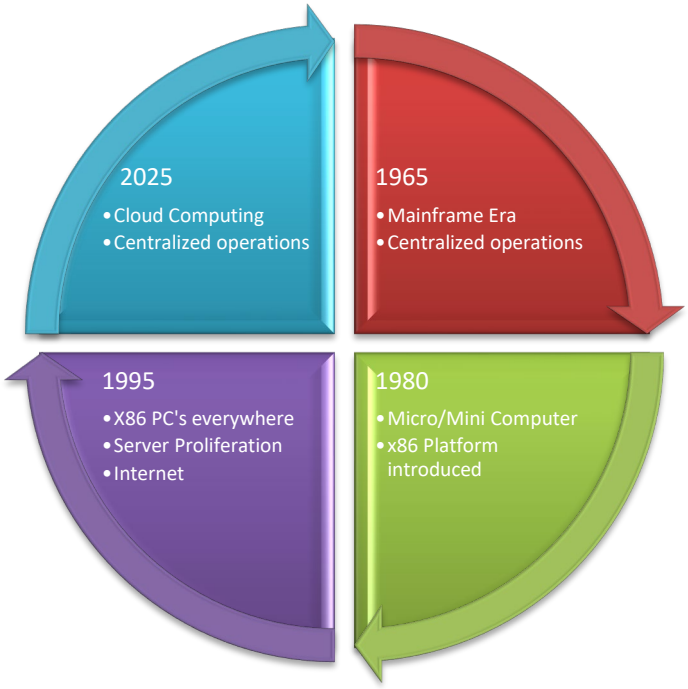
This work is offered as-is and as-available, with no warranties of any kind, whether express, implied, statutory, or other. This includes, without limitation, warranties of title, merchantability, fitness for a particular purpose, non-infringement, absence of latent or other defects, accuracy, or the presence or absence of errors, whether or not known or discoverable.

General Concepts

Cloud Computing

IT Lifecycle

In the last half century, computing has evolved full-circle. Where electronic computing began as a highly centralized resource, it has evolved to become a highly distributed and uncontrolled entity.



In 1965 computers weighed tons, occupied buildings, and cost millions

- Researchers at IBM invented the first hypervisor to apportion resources on expensive, highly centralized equipment.

By 1980, business and corporations could own and manage their own computers and manage their own resources!

- This era is hallmarked by the mini-computer and widespread availability of the micro-computer

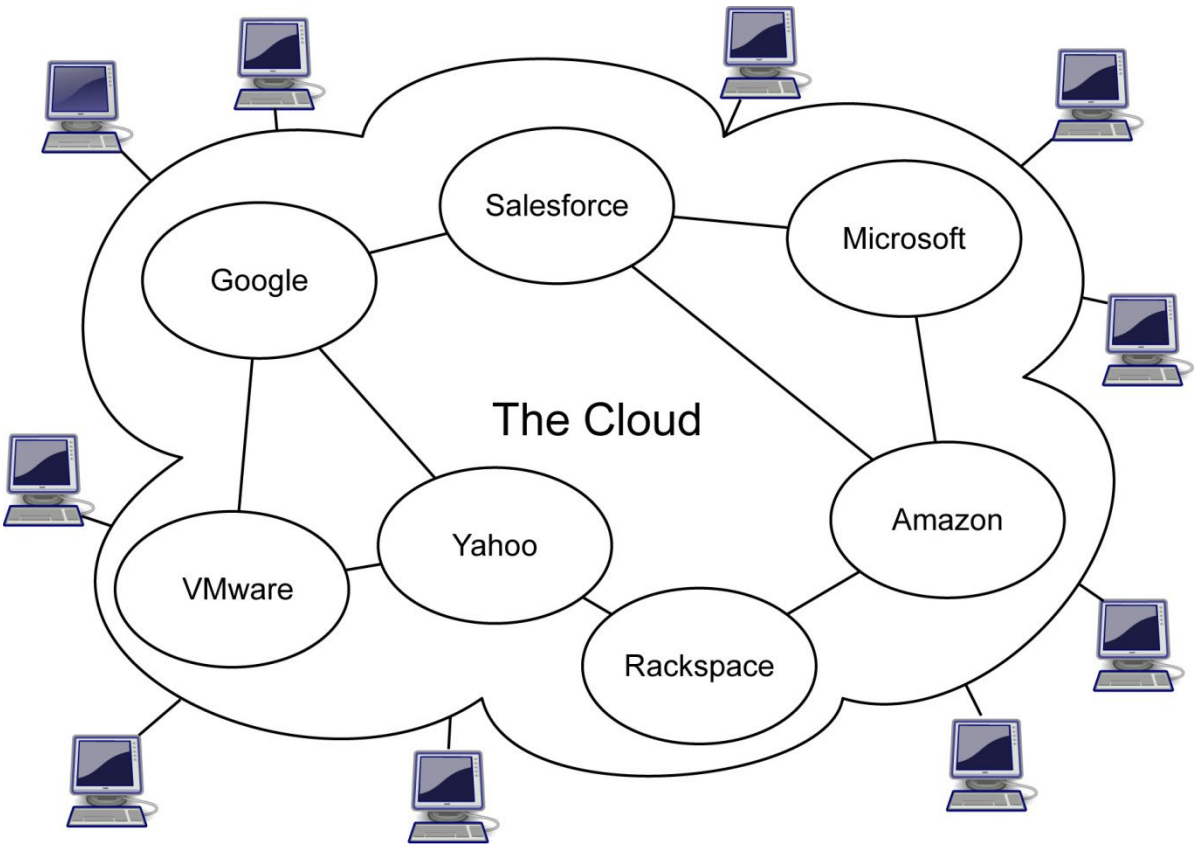
When 1995 rolled around, not only had the x86 revolution taken hold, but the internet had gone public, leading to massive proliferation of servers

- It was clear that a better solution was necessary

Now, in the 202X's, computing is on its way to being centralized again – with "servers" running on massively provisioned "nodes" which are located in Datacenters

- "Cloud Computing" is the re-mainframing of computing resources today!

The Difference Between Cloud Computing and Virtualization



Cloud computing is a paradigm shift following the switch from mainframe era of the 1960's to the client–server model beginning in the 1980s and continuing today. Resources are provisioned for users, who no longer have need for expertise in, or control over, the technology infrastructure "in the cloud" that supports them. Cloud computing describes a new delivery model for IT resources and services based on IEEE 802.3 Ethernet and the Internet, typically involving remote delivery of dynamically scalable and often virtualized resources. It is a byproduct and consequence of the ease-of-access to remote computing sites provided by the Internet.

In its simplest form, Cloud Computing takes the form of web-based programs or applications that users can access and use through a web browser as if it were a program installed locally on their own computer. The term "cloud" is used as a metaphor for IEEE 802.3 computer networks, based on the cloud drawing used in the past to represent the telephone network.[i]

- Virtualization = layer of abstraction between services and hardware
- Cloud Computing = layer of abstraction between consumers and services

Essential Characteristics of Cloud Computing[ii]

- On-demand self-service. A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider.
- Broad network access. Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations).
- Resource pooling. The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacenter). Examples of resources include storage, processing, memory, and network bandwidth.
- Rapid elasticity. Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time.
- Measured service. Cloud systems automatically control and optimize resource use by leveraging a metering capability1 at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service.

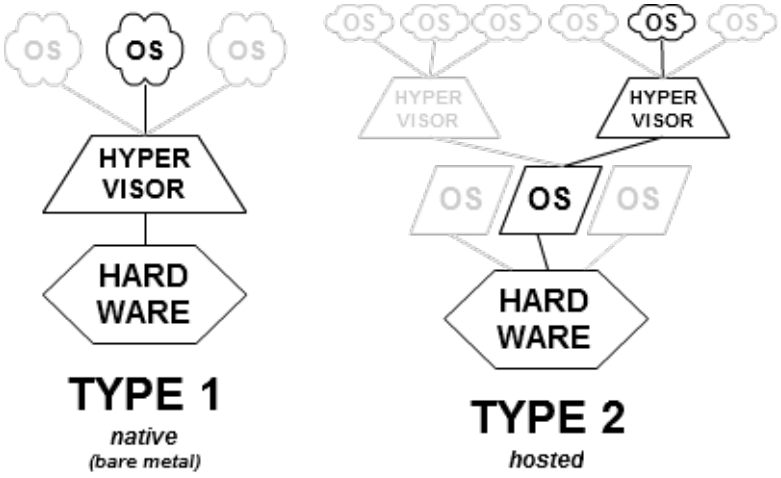
Or, put another way: all of the characteristics which also define virtualization!

Cloud Computing Service Models (Essential Terms)[iii]

- Software as a Service (SaaS). The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface, such as a web browser (e.g., web-based email), or a program interface. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.
- Platform as a Service (PaaS). The capability provided to the consumer is to deploy onto the cloud infrastructure consumer-created or acquired applications created using programming. The consumer does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly configuration settings for the application-hosting environment.
- Infrastructure as a Service (IaaS). The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include operating systems and applications. The consumer does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, and deployed applications; and possibly limited control of select networking components (e.g., node firewalls).

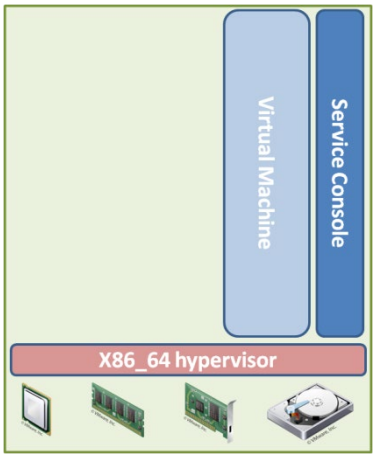
What is a Hypervisor?

A hypervisor is a computer software/hardware virtualization platform that allows multiple guest operating systems or containers to run concurrently on a single node.

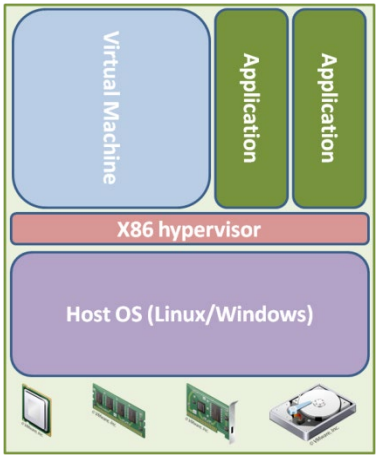


Hypervisor Classifications

Hypervisors are classified in two types:



- *Type 1* (*bare-metal*) hypervisors are software systems that run directly on the node's hardware and apportions the available hardware resources to guest Virtual Machines and/or containers, while maintaining isolation between the guests. The original hypervisor was developed at IBM in the 1960s. More recent examples are: VMware ESXi Server, Microsoft Hyper-V, Nutanix, and Proxmox.



- *Type 2* (or *hosted*) hypervisors are software applications running within a conventional operating system. In the case of Type 2 Hypervisors, the native Operating System runs on the hardware, the Hypervisor runs on (and claims its resources from) the node Operating System then presents those resources to Virtual Machines. Examples include: VMware Workstation, VMware Fusion

Proxmox Virtual Environment (PVE)

PVE is one of the most robust platforms available, and an ideal candidate for users looking to migrate from other Hypervisors.

PVE has the distinct advantage of being highly compatible with most hardware. It will install on existing servers, connect to existing SANs, and is natively capable of Hyperconverged storage with Ceph. PVE is also available in a variety of licenses from free (albeit without support or access to enterprise repos) to full enterprise support at a fraction of the cost of VMware under Broadcom.

Installing PVE is a very simple process. Since PVE has such a wide hardware compatibility, it will install on most servers, but it is always recommended to use the best/latest servers your organization can afford.

Furthermore, PVE supports both traditional SAN and Hyperconverged storage at no additional cost. That means you can use your almost certainly install PVE in place of other hypervisors, then gradually migrate your VMs from the other platform to PVE.

For the most part, managing PVE is simple and straightforward, using the embedded GUI. There are some instances where PVE will absolutely require the use of the CLI. In terms of features, PVE is comparable to all major Type-1 hypervisors, and superior to many. PVE supports the flowing features critical to many Organizations:

- High Availability failover of VMs
- Live Migration of VMs between PVE nodes
- iSCSI & FC SAN
- Hyperconverged storage
- Live patching of PVE nodes
- Supported by major backup platforms including Veeam

Why choose PVE over other platforms?

We consider PVE to be the best, most scalable and affordable platform available based on a comparison of features and attributes of other likely candidates.

Comparison of Hypervisor Platforms

Feature	Proxmox VE	Hyper-V	Nutanix	Scale Computing	VMware
Perpetual License	YES	YES	NO	NO	NO
Hypervisor Architecture	KVM	Hyper-V	KVM	KVM	VMware ESXi
Hypervisor Type	Type-1	Type-1	Type-1	Type-1	Type-1
Live Migration	YES	YES	YES	YES	YES
High Availability	YES	YES	YES	YES	YES
Native HCI	YES	YES (extra cost)	YES	YES	YES (extra cost)
SAN Storage	YES	YES	NO	NO	YES
Container Support	NATIVE	NO	EXTRA	NATIVE	EXTA
Pricing Basis	CPU	Cores	Cores & Capacity	Cores & Capacity	Cores & Capacity
Hardware Compatibility	WIDE	WIDE	LIMITED	LIMITED	LIMITED
Open Source	YES	NO	NO	NO	NO
License	GNU/GPL	PROPRIETARY	PROPRIETARY	PROPRIETARY	PROPRIETARY
Backup and DR support	WIDE	WIDE	WIDE	WIDE	WIDE
Native Backup and DR	YES	NO	YES	YES	NO
Native Platform Migration	YES	NO	YES	YES	YES

Licensing Considerations

PVE is GNU/GPL with perpetual licensing.

- Proprietary vs. GNU/GPL
    - As we all have seen through the Broadcom / VMware debacle, proprietary licensing means that the vendor can do anything they want regarding future licenses. They can even cut-off support & updates for the “perpetual” licenses you have purchased and threaten you with cease & desist orders if you have applied updates past the expiration of your most recent support/subscription contract.
    - GNU/GPL is a public license which does not expire. You will always be able to use and update what you have installed without penalty, even if you let support/subscription expire.
    - Additionally, the GNU/GPL license explicitly allows forking, dis-incentivizing vendors from applying unreasonable terms to licensed platforms.
- Perpetual vs. Term Licensing
    - Perpetual licensing refers to your ability to fully use a product (legally) after support/subscription contracts expire.

- With Term licensing, functionality of the product will likely be limited or cease to operate after support/subscription contracts expire.

Hardware Compatibility

Hardware Compatibility is another important consideration, especially for those considering a move from another Hypervisor. Many of the "proprietary silo" platforms have narrow hardware compatibility requirements, making them unsuitable for a direct replacement of another platform.

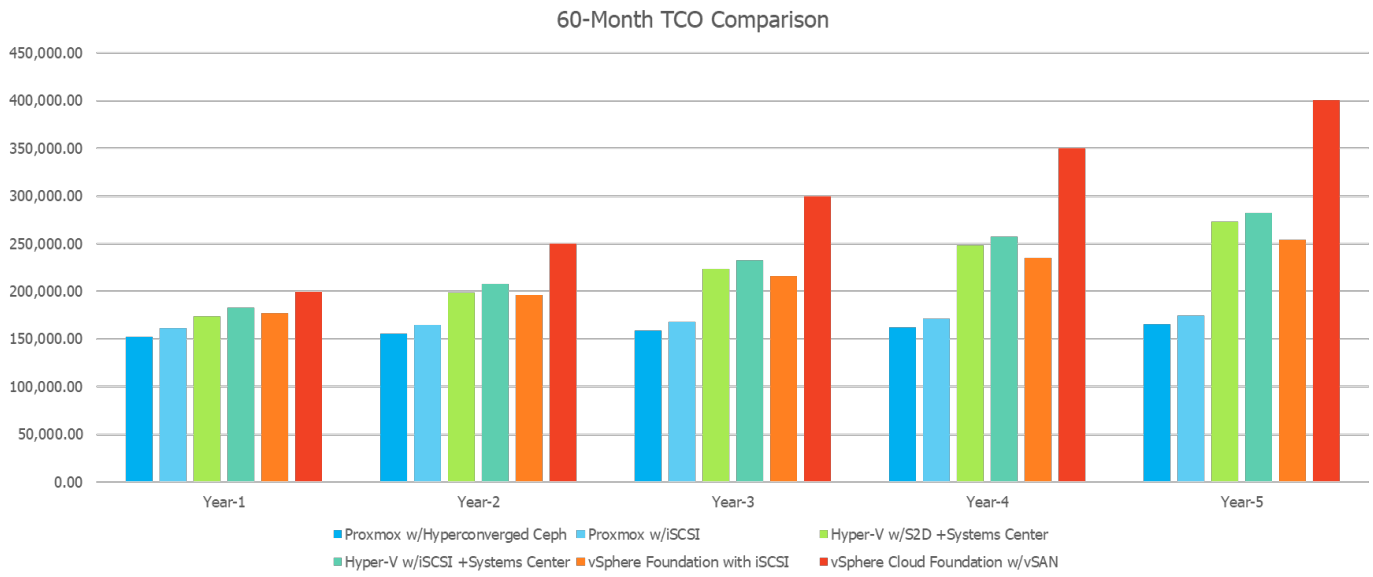
- Some platforms focus on selling "appliances" which are nothing more than highly-tuned nodes
- Other platforms only support specific "certified" devices (SCSI / SAS / NIC / PCI) and CPU generations

PVE has gone a different direction and worked around standards compatibility. That means that PVE will install on just about any modern server hardware and support what that hardware can do.

Hyperconverged vs. SAN Storage

While Hyperconverged storage is awesome, it does not always pay for itself. Moreover, if you are migrating from a platform with SAN storage, you will want to continue to leverage your investment in those SANs.

- Some alternative platforms don't support traditional SAN storage
- Other platforms charge you per capacity to use disks/hardware you have purchased



In the chart above pricing as of 2/1/2026, assumed is:

- $158,000 cluster investment w/SAN
- $149,000 cluster investment for HCI
- Published pricing as of 02/06/2026

Cluster Design for PVE

PVE has the distinct advantage of wide hardware compatibility. In most environments, PVE will install in place of existing Hypervisors and make use of existing storage.

Size of PVE Cluster

PVE clusters require a quorum of three nodes to function with Ceph and should be incremented in odd numbers. That may seem odd (pun intended), however the high availability mechanisms of PVE, called Corosync, function better with odd numbers to avoid a stalemate in electing failover nodes. If an even number of nodes are required, it is possible to run a witness Corosync node alongside the cluster using any hardware capable of supporting a Debian Linux OS.

We generally prefer smaller clusters with better provisioned hardware to larger clusters with less provisioning. This is due to the internal network speed of a server as opposed to the wire speed of your network.

The nature of Hyperconverged storage, however, may lead to the necessity for more nodes to accommodate disks and increase disk efficiency.

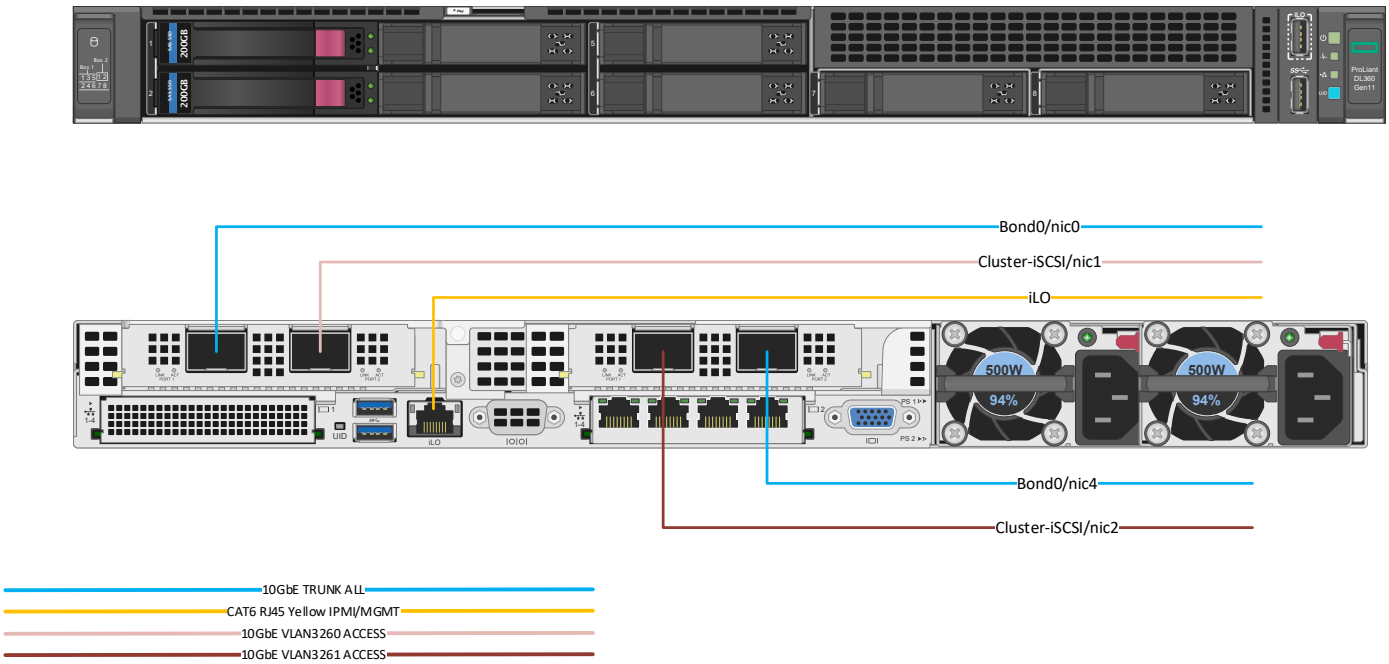
Server Nodes for PVE Cluster

As I said previously, one of PVE's distinct advantages is that it has wide hardware compatibility. It will install on most X86 servers built in the last 10 years. This makes it a perfect candidate for in-place conversion and migration from other Hypervisors.

Here are some basic recommendations/requirements for Proxmox server hardware:

- Minimum 4 NICs on 2 different PCI devices
- 2 Boot Disks in RAID 1 configuration
    - Small/inexpensive disks will do
- Storage disks for Ceph and ZFS
    - Disk controller must be able to present individual disks as 'HBA mode'
- CPUs depend on application requirements
    - Proxmox does not charge by core so high-core-count CPUs can work well
- RAM
    - Proxmox overhead is generally low in the 2-4% range

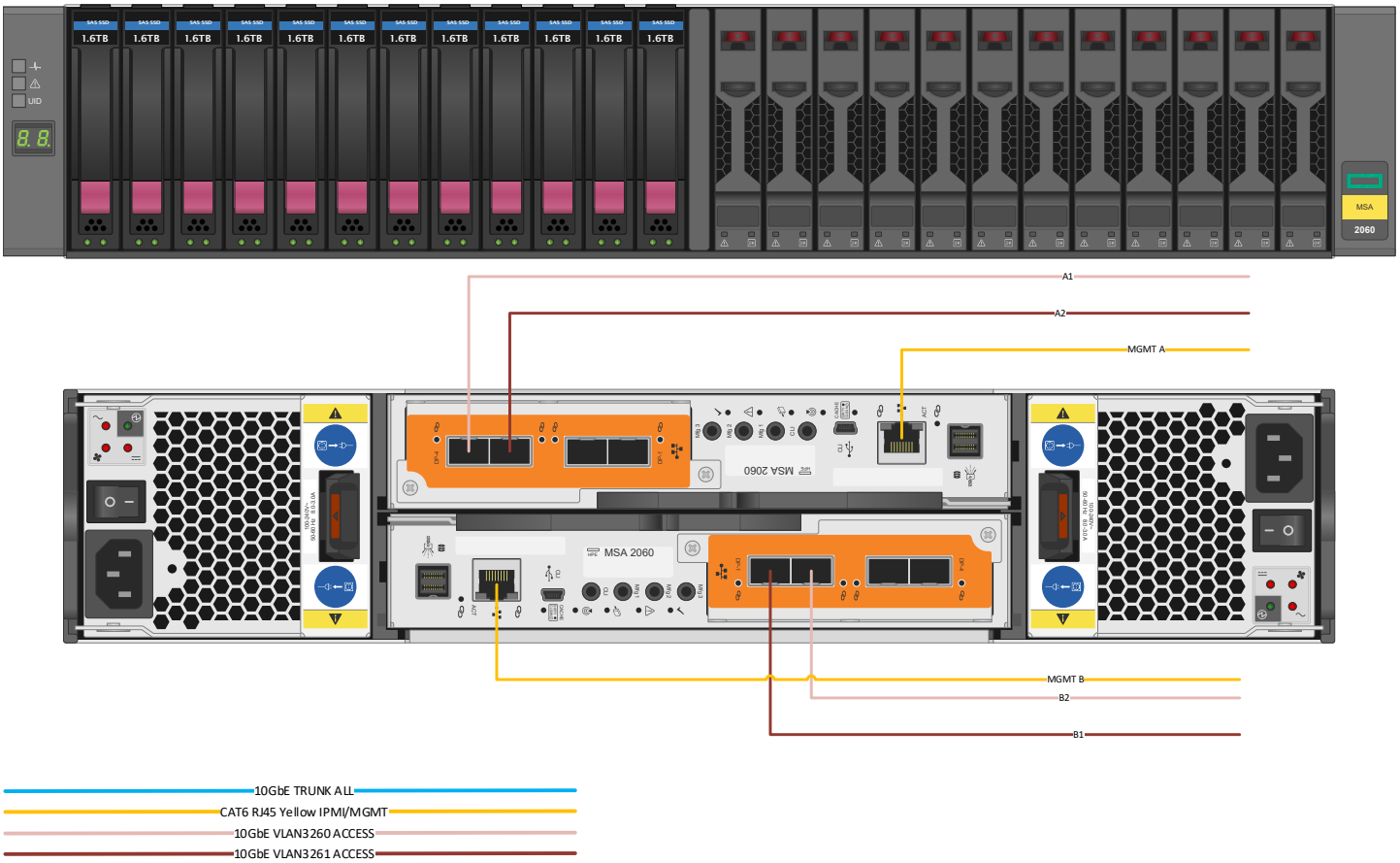
Node Design for PVE



Here you see a typical Proxmox server Node:

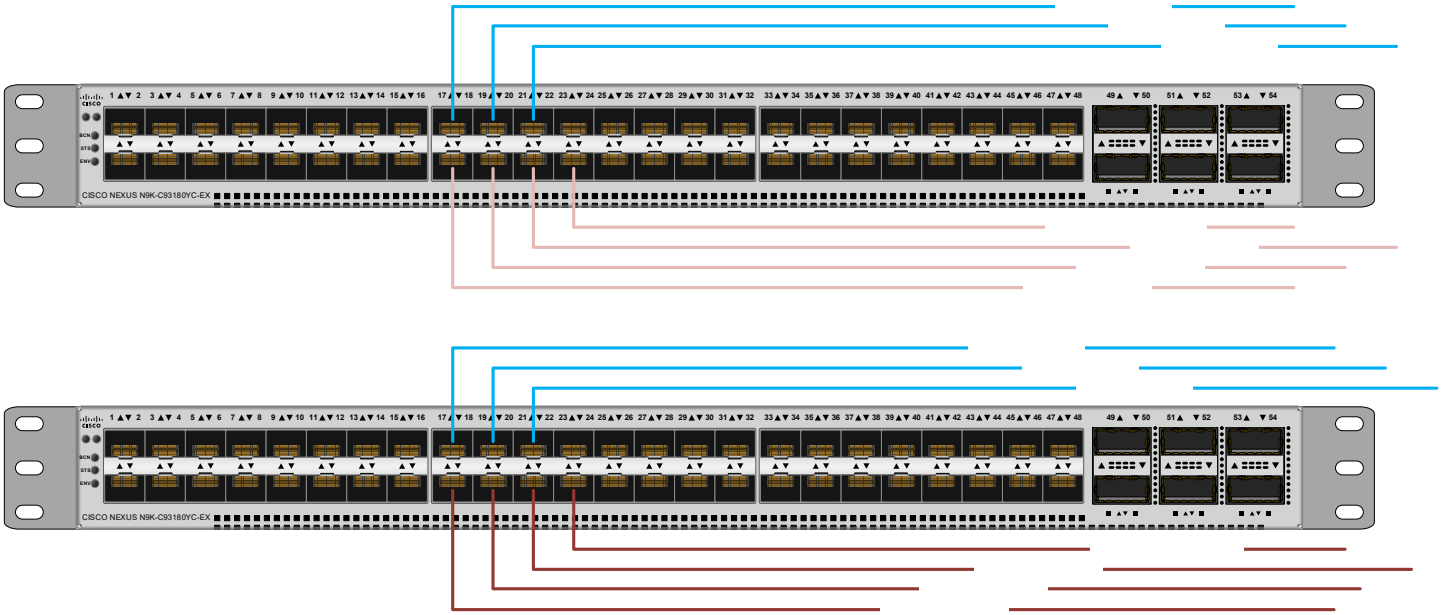
- Bond0 NICs are connected on 2 different PCI devices, which should be connected to two separate upstream Leaf/Rack Switches in TRUNK mode if VLANs are used
- Cluster/iSCSI has separate VLANs for each unique NIC in ACCESS mode

SAN Design for PVE



- When iSCSI is configured for Proxmox, Best Practice is to separate paths on different Networks.

Network Switch Design for PVE



- All TRUNK connections distributed across two infrastructure switches
- SAN is connected to two infrastructure switches

Types of Storage for PVE

The first thing to consider with a PVE installation or migration is the desired target storage. Here again, PVE shines as having wide compatibility with most storage devices including: local disks, SAN, NAS and Hyperconverged storage.

Local Disks

Local disks are fine for standalone installations and can be leveraged as part of a Hyperconverged cluster. Local storage is generally fast and reliable, but unless it is clustered, it is not Highly Available (HA) and will not support live migration or fail-over of workloads.

NAS/NFS

A NAS (Network Attached Storage) is a device that presents file-level storage using the NFS protocol That is to say that your workloads are stored on the native filesystem of the NAS device. Windows and Linux are both capable of presenting NFS storage, and in both cases, workloads would be stored on the respective underlying filesystem: NTFS, ReFS, ext3, ext4, etc.

NAS systems create High Availability through multiple network paths between server and storage and may also have multiple, redundant controllers to create those paths.

NAS can support both live migration and fail-over of workloads. NFS is generally thin-provisioned on the target filesystem, however pre-allocation may prevent you from using more than the total of all your workloads committed space.

SAN

A SAN (Storage Attached Network) is a device that presents block-level data to initiators using primarily iSCSI or FC protocol. That is to say that the SAN presents unformatted “disks” as LUNs (Logical Unit Number) to servers, which then format the LUNs and consume them.

SANs create High Availability using redundant hardware controllers and multiple pathways between server and storage.

SANs support live migration and fail-over of workloads. At the moment, PVE does not support thin-provisioning of SAN workloads in a clustered environment due to the limitations of LVM (Logical Volume Manager) volumes.

Hyperconverged Storage with Ceph

Ceph is a resilient and widely used Hyperconverged storage platform which aggregates the local disk space across multiple nodes to create a pool of Highly Available storage. Ceph does not rely on any other filesystem but instead manages individual disks with its own storage backend called BlueStore.

A cluster of nodes which deploy Ceph storage can tolerate one or more node failures, configurable in the basic Ceph settings.

Ceph supports live-migration and fail-over of workloads. Ceph is also natively thin-provisioned.

Key advantages of Ceph include:

- Unified Storage: Ceph provides object, block, and file storage in one platform, eliminating the need for separate systems.
- Massive Scalability: Ceph can grow from a few nodes to thousands, scaling to petabytes or exabytes without performance degradation.
- Cost-Effective: As open-source software that runs on commodity (off-the-shelf) hardware, it eliminates vendor lock-in and high licensing costs, reducing total ownership costs.
- High Availability & Self-Healing: Using the CRUSH algorithm, Ceph eliminates single points of failure, automatically rebalancing and repairing data (self-healing) if a node fails.
- No Metadata Bottleneck: The CRUSH algorithm allows clients to calculate data locations rather than querying a central server, ensuring high performance.
- Versatile Use Cases: Ideal for cloud infrastructure (OpenStack, Kubernetes), big data analytics, and backup, providing flexibility for diverse workloads.
- Enterprise Features: Includes advanced capabilities like thin provisioning, snapshots, and disaster recovery replication.

Comparison of common PVE storage subsystems

	Live Migration	Fail-over	Thin-provisioning
Local Disks	NO	NO	YES
NAS/NFS	YES	YES	YES
SAN	YES	YES	NO
Ceph	YES	YES	YES

Table 1

Considering Hyperconverged Storage

Hyperconverged storage is not appropriate for every deployment. The common myth that Hyperconverged storage reduces the number of devices in your environment is simply not true.

While Hyperconverged storage eliminates the need for a traditional SAN or NAS (a device), it will probably require a larger number of individual disks; for smaller clusters it will be a lot more. Given the cost of fast enterprise-grade SSDs or NVMe "disks," Hyperconverged storage may or may not be more affordable. The fact that Ceph is included with every licensed level of PVE, definitely makes Hyperconverged storage a greater possibility.

Traditional SAN vs. Hyperconverged Storage

From a performance standpoint, we consider the difference between a traditional SAN and Hyperconverged storage a wash. Both depend on: The IOPS (Input/Output per Second) of each disk they contain, network speed and controller capability/cache.

Garbage in = Garbage out; whereas if you provision fast SSDs or NVMEs on high-performance controllers, either SAN or Hyperconverged storage is capable of great performance.

In the tables below, you can see that in a 3-node cluster, Hyperconverged storage requires many more disks to achieve equal NET storage capacity. As cluster size grows, both disk efficiency and performance of Hyperconverged storage increases. In a cluster of 4 nodes, for example, PFTT=2 would achieve a NET disk use efficiency of 50%.

Disk Requirements for Hyperconverged Storage

3-Node Hyperconverged Cluster / PFTT =2				
3.84 TB	3.84 TB	3.84 TB	3.84 TB	
3.84 TB	3.84 TB	3.84 TB	3.84 TB	
3.84 TB	3.84 TB	3.84 TB	3.84 TB	92.16 TB RAW
3.84 TB	3.84 TB	3.84 TB	3.84 TB	30.72 TB NET
3.84 TB	3.84 TB	3.84 TB	3.84 TB	
3.84 TB	3.84 TB	3.84 TB	3.84 TB	

Table 2

Disk Requirements for Traditional SAN

Traditional SAN / RAID 6 (N-2)					
3.84 TB	3.84 TB	3.84 TB	3.84 TB	3.84 TB	38.4 TB RAW
3.84 TB	3.84 TB	3.84 TB	3.84 TB	3.84 TB	30.72 TB NET

Table 3

General Networking Concepts

We need to spend some time discussing general networking concepts before diving into PVE, because PVE is much more modern in its approach to networking than some other platforms.

Windows (Hyper-V) is particularly oriented around its origins with Classful networking, and even VMware has some tendency to stick to the IP/Subnet Mask relationship.

In almost every case, PVE is going to require the use of CIDR (Classless Inter-Domain Routing) notation when entering or creating IP addresses.

Classful Networks

Full networks are divided into class A, B and C, depending on the size (number of possible addresses) of each network. Classful networking is considered obsolete following the introduction of Classless Inter-Domain Routing (CIDR) in 1993[iv]

	CIDR	SNM	Example	First usable IP	Last usable IP
Class A	/8	255.0.0.0	10.0.0.10	10.0.0.1	10.0.0.254
Class B	/16	255.255.0.0	172.16.0.10	172.16.0.1	172.16.255.254
Class C	/24	255.255.255.0	192.168.0.10	192.168.0.1	192.168.0.254

Table 4

CIDR (Classless Inter-Domain Routing) / SNM

CIDR was introduced in 1993 and allows for the almost infinite subnetting (dividing into smaller network portions) of available IP spaces.

Possible CIDR Networks

CIDR	Subnet Mask	Example Subnet	First IP	Last IP	IP Addresses
/32	255.255.255.255	10.0.0.0/32	10.0.0.0	10.0.0.0	1
/31	255.255.255.254	10.0.0.0/31	10.0.0.0	10.0.0.1	2
/30	255.255.255.252	10.0.0.0/30	10.0.0.0	10.0.0.3	4
/29	255.255.255.248	10.0.0.0/29	10.0.0.0	10.0.0.7	8
/28	255.255.255.240	10.0.0.0/28	10.0.0.0	10.0.0.15	16
/27	255.255.255.224	10.0.0.0/27	10.0.0.0	10.0.0.31	32
/26	255.255.255.192	10.0.0.0/26	10.0.0.0	10.0.0.63	64
/25	255.255.255.128	10.0.0.0/25	10.0.0.0	10.0.0.127	128
/24	255.255.255.0	10.0.0.0/24	10.0.0.0	10.0.0.255	256
/23	255.255.254.0	10.0.0.0/23	10.0.0.0	10.0.1.255	512
/22	255.255.252.0	10.0.0.0/22	10.0.0.0	10.0.3.255	1,024
/21	255.255.248.0	10.0.0.0/21	10.0.0.0	10.0.7.255	2,048

CIDR	Subnet Mask	Example Subnet	First IP	Last IP	IP Addresses
/20	255.255.240.0	10.0.0.0/20	10.0.0.0	10.0.15.255	4,096
/19	255.255.224.0	10.0.0.0/19	10.0.0.0	10.0.31.255	8,192
/18	255.255.192.0	10.0.0.0/18	10.0.0.0	10.0.63.255	16,384
/17	255.255.128.0	10.0.0.0/17	10.0.0.0	10.0.127.255	32,768
/16	255.255.0.0	10.0.0.0/16	10.0.0.0	10.0.255.255	65,536
/15	255.254.0.0	10.0.0.0/15	10.0.0.0	10.1.255.255	131,072
/14	255.252.0.0	10.0.0.0/14	10.0.0.0	10.3.255.255	262,144
/13	255.248.0.0	10.0.0.0/13	10.0.0.0	10.7.255.255	524,288
/12	255.240.0.0	10.0.0.0/12	10.0.0.0	10.15.255.255	1,048,576
/11	255.224.0.0	10.0.0.0/11	10.0.0.0	10.31.255.255	2,097,152
/10	255.192.0.0	10.0.0.0/10	10.0.0.0	10.63.255.255	4,194,304
/9	255.128.0.0	10.0.0.0/9	10.0.0.0	10.127.255.255	8,388,608
/8	255.0.0.0	10.0.0.0/8	10.0.0.0	10.255.255.255	16,777,216
/7	254.0.0.0	10.0.0.0/7	10.0.0.0	11.255.255.255	33,554,432
/6	252.0.0.0	8.0.0.0/6	8.0.0.0	11.255.255.255	67,108,864
/5	248.0.0.0	8.0.0.0/5	8.0.0.0	15.255.255.255	134,217,728
/4	240.0.0.0	0.0.0.0/4	0.0.0.0	15.255.255.255	268,435,456
/3	224.0.0.0	0.0.0.0/3	0.0.0.0	31.255.255.255	536,870,912
/2	192.0.0.0	0.0.0.0/2	0.0.0.0	63.255.255.255	1,073,741,824
/1	128.0.0.0	0.0.0.0/1	0.0.0.0	127.255.255.255	2,147,483,648
/0	0.0.0.0	0.0.0.0/0	0.0.0.0	255.255.255.255	4,294,967,296

Table 5

Private IPv4 Networks (RFC 1918)

Private networks came about in the form of RFC 1918 published by the Internet Engineering Task Force (IETF). Private networks are largely a reaction to the diminishing number of available IPv4 addresses on the Internet.

Be designating certain network ranges as private (not publicly routable), RFC 1918 allows organizations to manage networks of varying sizes internally and then make services on those network(s) available to the Internet using NAT

CIDR	RFC 1918 name	IP address range	Largest CIDR block (subnet mask)
/8	24-bit block	10.0.0.0 – 10.255.255.255	10.0.0.0/8 (255.0.0.0)
/12	20-bit block	172.16.0.0 – 172.31.255.255	172.16.0.0/12 (255.240.0.0)
/16	16-bit block	192.168.0.0 – 192.168.255.255	192.168.0.0/16 (255.255.0.0)

Table 6

Subnetting

Subnetting is the process of dividing a larger network into smaller portions. Almost all the private networks we use and deploy are subnets of a larger network.

A small organization with one site might adequately get by with just a few subnets of the private192.168.0.0/16 network. E.g.: 192.168.0.0/24 and 192.168.1.0/24 This is obviously limited to 254 distinct subnets.

A medium size organization requiring more distinct networks might choose to subnet the private 172.16.0.0/12 network. E.g.: 172.16.0.0/24 and 172.16.1.0/24 This provides more possible distinct subnets, however, there is no elegant way of designating site/VLAN in this plan.

How Class A /24 Subnets Work

We prefer to subnet the 10.0.0.0/8 private network. In this way we achieve elegant distinction of site/VLAN/host.

A common and recommended practice is using the second part of a class A network (second octet) to designate the location, the third octet to designate the VLAN and the last octet as the host IP. This provides an elegant and simple solution.

Our training class (if you use our equipment/labs) will operate in our location 26, using VLANs 20-26, with each network subnetted into segments of 254 host IP addresses (/24).

Class A	Location	VLAN	HOST
10	26	20	101

Table 7

/24 Network Reservations

For most of us, some form of planned organization within a subnet/network is required. We organize our /24 networks this way:

Network services	X.Y.Z.1 to X.Y.Z.9	Gateways / DHCP / PXE
Domain Services	X.Y.Z.10 to X.Y.Z.19	DCs / WSUS / KMS
Static IP	X.Y.Z.20 to X.Y.Z.ABC	Servers
DHCP POOL	X.Y.Z.ABD to X.Y.Z.254	Optional DHCP pool

Table 8

Using This Book

We've made every attempt to make this book as readable as possible so you may continue to use it for future reference long after the class is completed. It is important to remember that not every class or situation is alike, and the screenshots/instructions contained on subsequent pages represent a baseline that works in our lab but will need to be adapted for different conditions.

The Materials Contained Within Are Cumulative

This book is arranged in as logical an order as we could conceive. We have used the very methodology that we use when building any new environment from the ground up in a production environment.

Each section is designed to introduce you to one of the building blocks of virtualization with Proxmox. Sections are arranged in order so that the knowledge gained in one section contributes to your understanding of the next. In this manner, rather than focusing on memorization of requirements and dependencies for various components of virtualization, your knowledge of requirements will be natural and fluent.

For example:

> **If** we taught you about the requirements for Live Migration using printed materials and a lecture, then connected to a pre-configured environment to configure and practice Live Migration; your knowledge of Live Migration dependencies would come from *memorization*.

In practice:

> We prefer to have you configure each and every one of the dependencies for a complete and functioning Proxmox environment in order. By following this methodology, every participant will be *acutely aware* of the interrelations and dependencies of all the Proxmox components and features! *Without memorization*, participants will inherently know and understand how Proxmox works.

A Few, Simple Rules....

- You must participate in all labs. Failure to do so will render your environment useless and you will not be able to complete class.
- You must follow the conventions established in this book or stated by your instructor. Failure to do so will invalidate your configuration and may inhibit other participants learning process.
- If you cannot complete a lab, merely sit-it-out and wait for the instructor to be available to assist you with completion.

Ways to learn PVE

Instructor-Led Training (ILT)

Instructor-led classes (either online or in a conventional classroom) represent most of the use for this book. The difference between a good instructor and a mediocre instructor is the individual's ability to answer questions which range beyond the scope of the material.

In tackling questions and subjects which may not, specifically, be covered by the material, it is often necessary, or desirable, to add or modify the default setup of the labs. During Instructor-led classes, pay attention to the variables that the instructor gives you for the completion of the lab. Do not simply complete all the labs with the default values provided in this book; to do so would, at best, abbreviate your learning experience, and at worst nullify subsequent labs for you and others.

Instructor-Led Training Resources

During an ILT class on our environment we allocate resources to each participant.

Each participant will be allocated:

1. 1 PVE node
    a. 64GB RAM
    b. (12) CPU
    c. Install disk
    d. (4) 300GB Ceph disks
2. 1 iSCSI LUN for the classroom PVE cluster
3. 1 iSCSI LUN per participant for each persons Proxmox Backup Server

The class will have access to 10GbE network, iSCSI SAN, AD domain, SMB and NFS shares. We have gotten all the technical pre-requisites takes care of so you can concentrate on learning PVE.

Self-Study

Participants wishing to engage in self-study will need to rigidly adhere to the conventions established in this book or create their own conventions (using their own lab equipment) entirely.

Please remember that while self-study and our instructor-led programs use the same book and materials, the very nature of an instructor-led class leads to change and adaptation throughout the session. You simply cannot participate in a group learning experience without participating with the group!

Self-Study Resources

To adequately provision your own lab for learning PVE, we recommend the following:

- PVE node server or VM: Minimum 16GB RAM (3 required)
- iSCSI SAN: We recommend TrueNAS for lab environments
- Network: 10GbE (best) but 1GbE will do

Our LAB

Through this process, we have done our best to emulate a typical Enterprise environment. Sure, you could create a simpler lab environment, but that would not provide the experience or knowledge you need at work.

We have configured and use:

- Windows DC
- Windows DNS
- Windows CA
- SMB Share
- VLANs
- iSCSI SAN
- Jumbo Frames
- 10GbE network
- L3 Routing/firewall
- OpenVPN for lab access

Each participant node has:

- 12 CPU
- 64 GB RAM
- 32 GB install disk
- (4) 300 GB storage disks (for use with Ceph)
- (6) 10GbE interfaces

We are big fans of Enterprise open-source, however we have chosen to rely on Windows and Microsoft DC/DNS/CA because that's what's used in real-world Enterprise environments. Sure, you could use open-source utilities to execute all the Infrastructure services (and it would work well), however our goal is to provide a relatable training experience that can journal directly to Organizational IT.

The Domain

To expedite the learning process, we have a pre-configured Active Directory domain configured:

- lab.vmsources.com

If you are using our resources for the learning process, you must use this domain and respect its conventions.

If you have provisioned your own resources for the learning process using, for example, VMware Workstation; you will be responsible for the independent acquisition of all binaries and correct configuration of an Active Directory domain

The Variables

My *Student Number* is:_____

My Lab Variables	
My *Student Number* is:	
XYZ=	
(ABC=XYZ+20) ABC=	
(DEF=XYZ+40) DEF=	
(GHI=XYZ+60) GHI=	
(JKL=XYZ+80) JKL=	

We use the default variable of XYZ for each participant in our lab environment. Subsequent variables will be ABD, DEF, GHI, JKL which are always separated by 20. You will be issued your own unique XYZ variable which will determine all unique lab variables such as names and IP addresses.

These variables, which are always bolded and in CAPS:

- XYZ is a number between 101 and 120, and will become the primary basis for all NAMES you use
- ABC is a number between 121 and 140 (always 20 greater than XYZ)
    - used in labs which require more than one unique resource
- DEF is a number between 141 and 160 (always 40 greater than XYZ)
    - used in labs which require more than two unique resources
- GHI is a number between 161 and 180 (always 60 greater than XYZ)
    - used in labs which require more than three unique resource
- JKL is a number between 181 and 200 (always 80 greater than XYZ)
    - used in labs which require more than three unique resource

Users and Passwords

There are pre-configured users on the domain.

- Username: adminXYZ@lab.vmsources.com (LAB\adminXYZ)
- Password:

Classroom Share

We have provisioned a SMB share for use by participants in our lab environment. The share will have all the files and ISOs we will use during the execution of these SBS LABs

- SMB://share/files
- Windows: \\share\files

Our Networks and VLANs for this Lab Environment

We will use several networks in executing labs. In class, we use VLANs and practice full network separation, as you would in any enterprise environment.

NOTE: If you are executing these labs on your own, it is possible to practice network separation without using VLANs (some home-lab networks/hypervisors do not support VLANs), simply use the interface designations and omit the VLAN ID. The IP/SNM will separate the networks functionally, albeit without true security.

VLAN	Designation	Network	Gateway	DHCP	MTU
20	PVE MGMT	10.26.20.0/24	10.26.20.1	NO	1500
21	iSCSI Backup	10.26.21.0/24	N/A	NO	1500
22	iSCSI 1	10.26.22.0/24	N/A	NO	9000
23	iSCSI 2	10.26.23.0/24	N/A	NO	9000
24	VMs	10.26.24.0/24	10.26.24.1	YES	1500
25	VDI VMs	10.26.25.0/24	10.26.25.1	YES	1500
26	VMware MGMT	10.26.26.0/24	10.26.26.1	NO	1500

Table 9

Our Interface Mapping for this Lab Environment

XYZ represents your participant number

Interface MAC	Pinned Interface name	Available VLANs	Mode
00:50:56:XY:Z0:01	nic1	20-26	TRUNK
00:50:56:XY:Z0:02	nic2	20-26	TRUNK
00:50:56:XY:Z0:03	nic3	20-26	TRUNK
00:50:56:XY:Z0:04	nic4	20-26	TRUNK
00:50:56:XY:Z0:05	nic5	22	Access
00:50:56:XY:Z0:06	nic6	23	Access

Table 10

Installing and Configuring PVE

SBS LAB – (GUI) Installing PVE

Unfortunately, if you are taking this class remotely, you will not be able to install PVE yourself. Your PVE server will be installed for you and ready for connection at its designated IP address: 10.26.20.XYZ:8006

The process of installing PVE is very easy as you can see in the screenshots below and class videos.

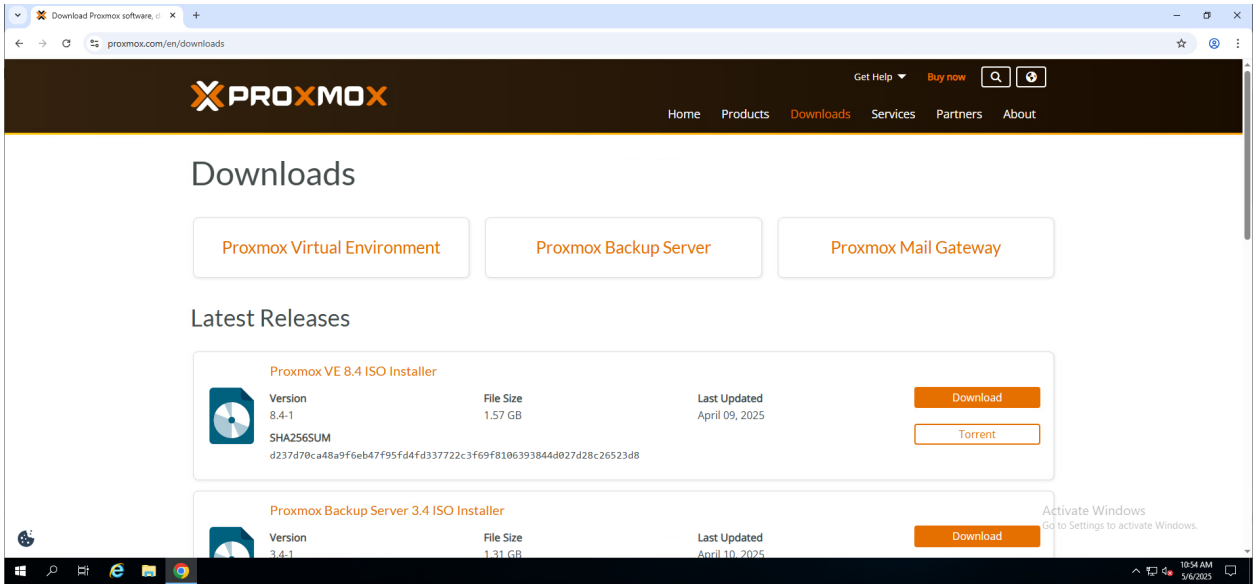
Variables for this lab (or other as appropriate for your environment):

- Username: root
- Password: P@ssw0rd
- Install disk: /dev/sda
- Country: United States
- Time zone: UTC
- Language: English
- Email: admin@lab.vmsources.com
- VLAN: 20
- IP: 10.26.20.XYZ/24
- Gateway: 10.26.20.1
- DNS: 10.26.24.10

1. First, we must get PVE

   Step 1: Download the PVE ISO: https://www.proxmox.com/en/downloads

   e.g.:



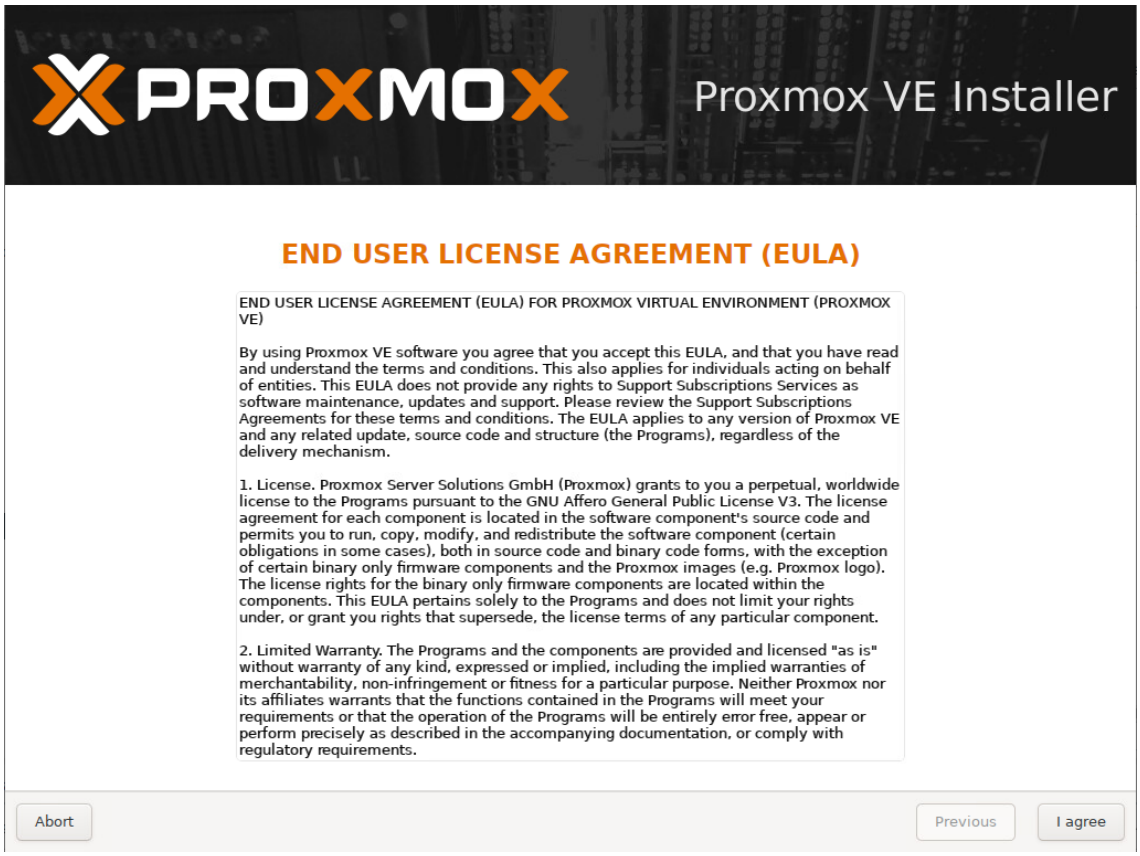
Step 2: Mount the ISO to the server you will be installing to and boot to ISO:

Step 3: Power-on the server and observe the console/KVM

e.g.:

Step 4: Follow the prompts in the installer. Select: Agree

e.g.:

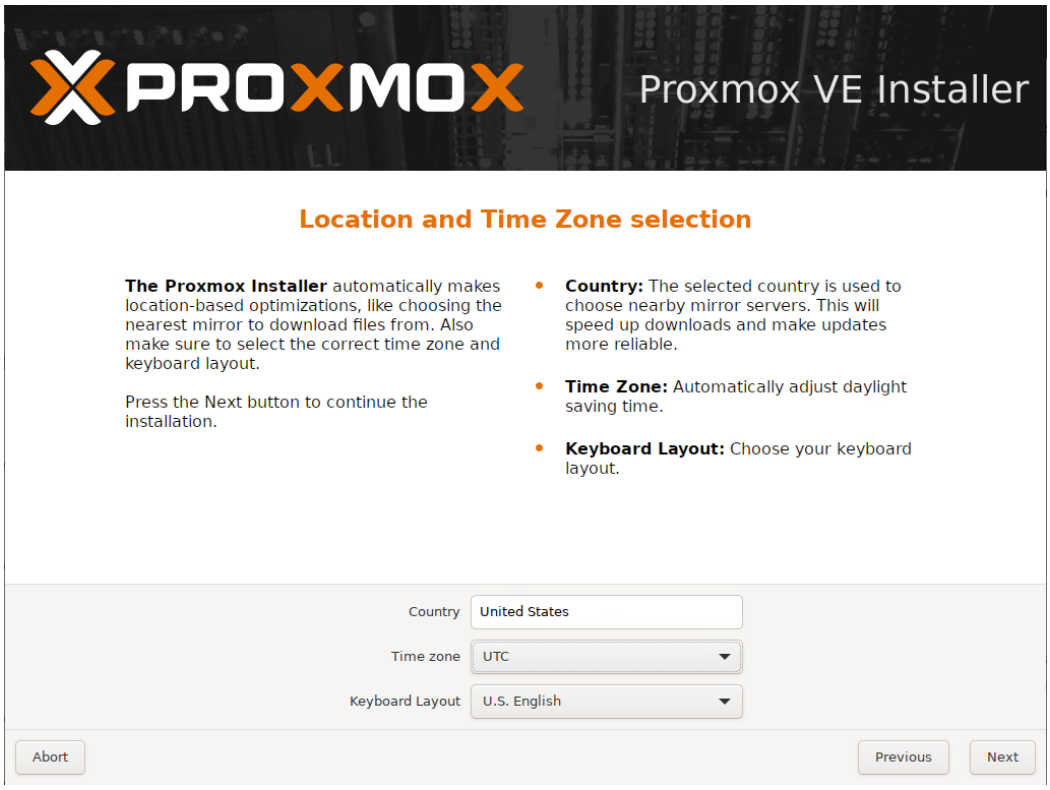


Step 5: Select the install disk, PVE can be installed to flash-media, but it is recommended to install PVE on redundant disks (RAID 1 or equivalent) such as HDD, SDD or NVMe (usually: /dev/sda)

e.g.:

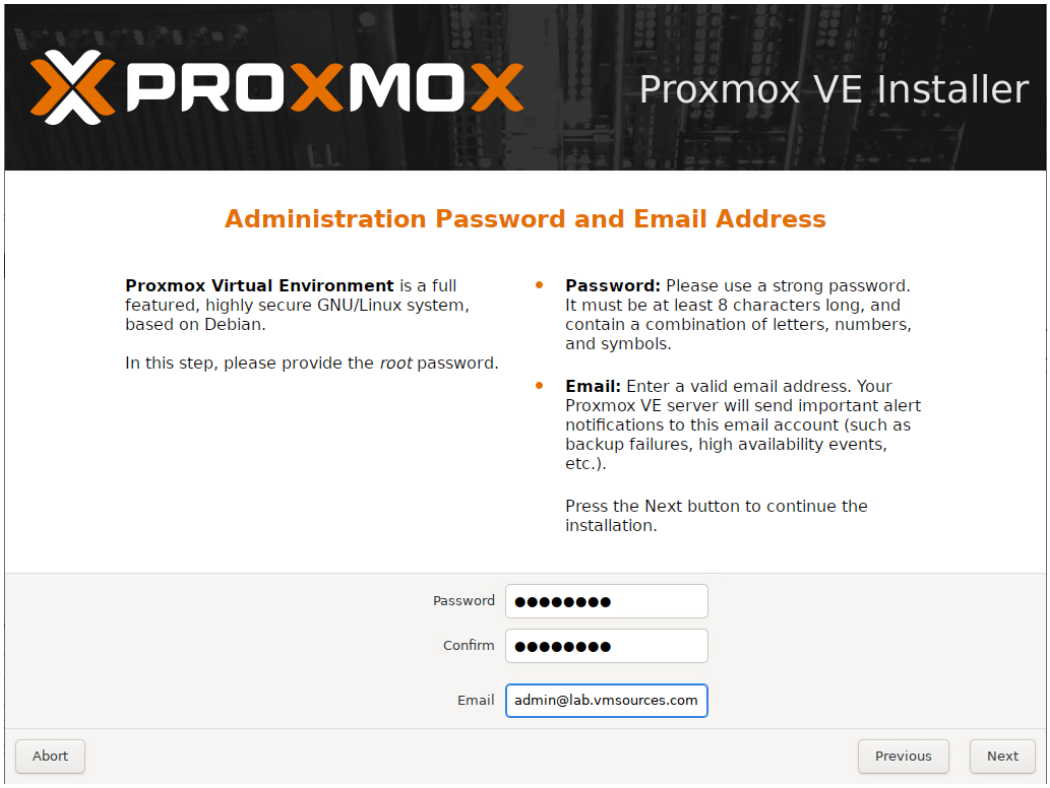
Step 6: Select your country, time zone and language. **We recommend using UTC for all infrastructure servers.**

e.g.:



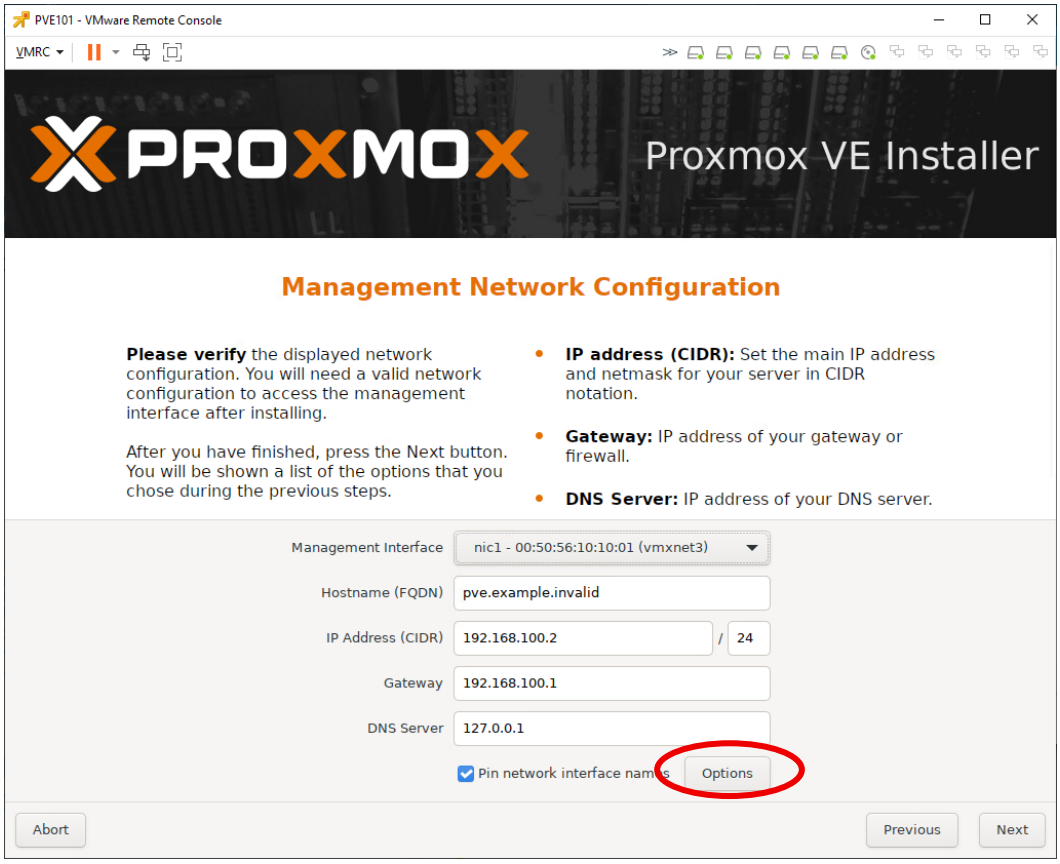
Step 7: Set password and email.

e.g.:



Step 8: Set the initial network properties for this PVE node. Start by clicking: Options

e.g.:

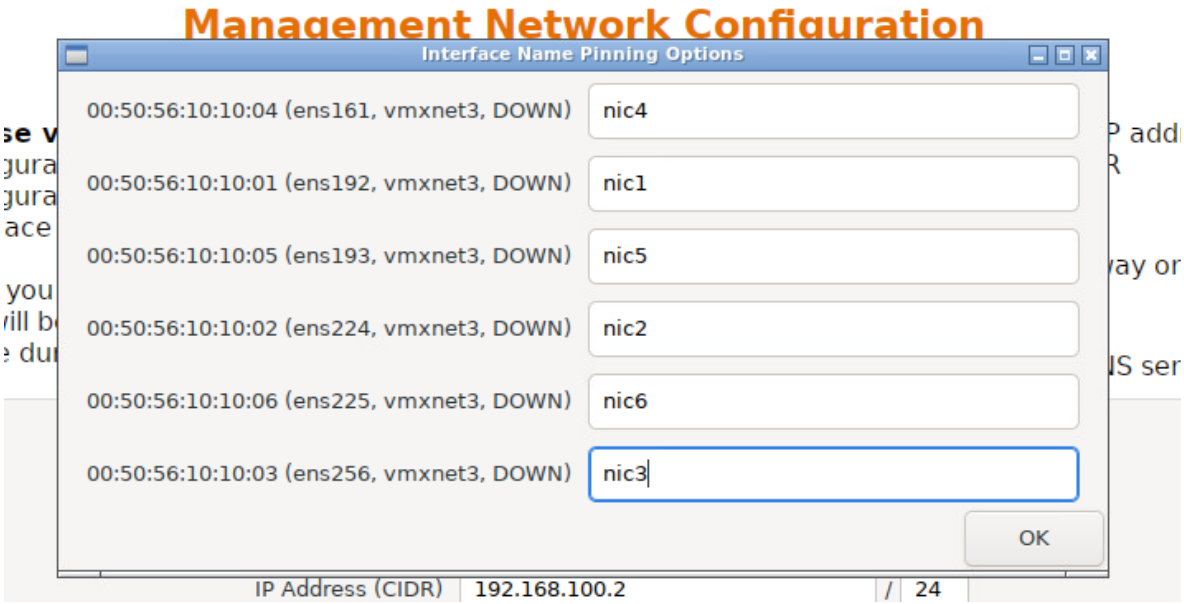


NOTE : How do you know what interface to use for management? You don't really, at least not until PVE is installed. If you have six equivalent interfaces, you could leave just the ones you intend to use for management connected (and they would have green dots to the left of the interface name), however after install is complete we can determine the MAC/Interface relation with the command: ip -a

NOTE : Pin network interface names is very useful so that the interfaces are named "nic1,nic2, etc." and not named with the hardware interface references, eg. "enp4s0"

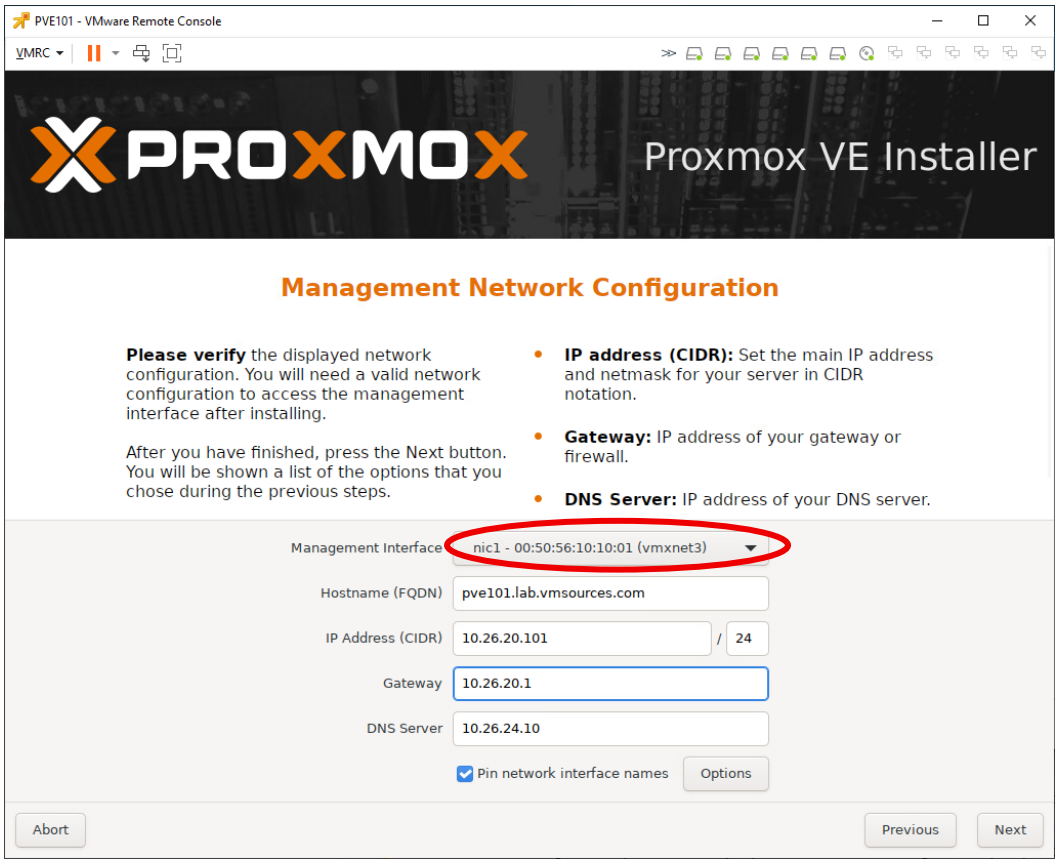
Step 9: We will want to align our Network Interface Names with Table 10, to make management easier in the future.

e.g.:



Step 10: Now assign network properties as shown, except substituting your unique values for the XAY/101 variables shown. Be sure to use nic1 as the Management Interface

e.g.:



Step 11: Install

e.g.:

2. Install complete.

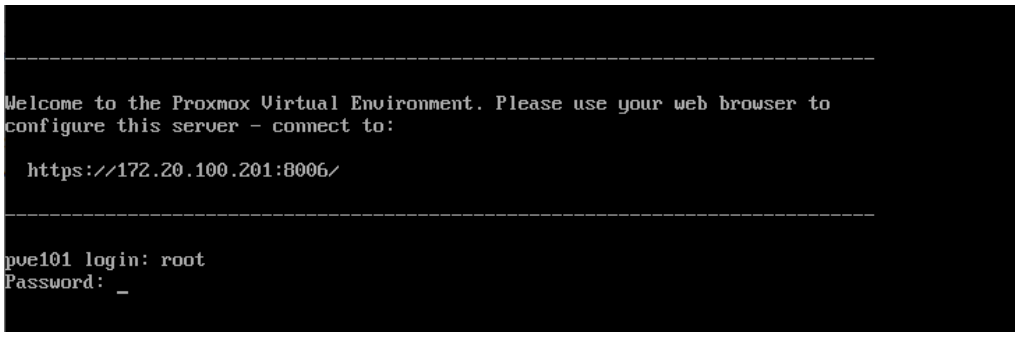
SBS LAB – (CLI) Initial configuration of PVE for VLANs

PVE provides no user-friendly VLAN option on installation, so if your environment uses VLANs, it will be necessary to edit the /etc/network/interfaces file a little bit

1. Now we log on to our PVE node

   Step 1: Login as root with the password you set.

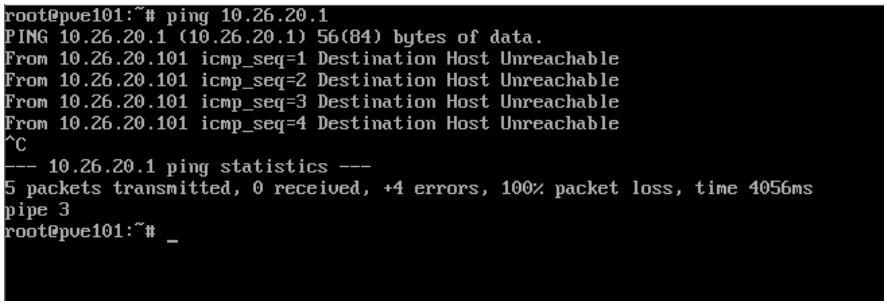
   e.g.:



   Step 2: Test networking by pinging the gateway or other known good IP:

   Command #1 run: ping 10.26.20.1

   e.g.:



   NOTE : As you can see, network connectivity is not presently functioning because our environment uses TRUNK ports for everything except iSCSI storage. We will need to configure our VLAN at the CLI to proceed

3. Determine NIC (MAC) relationship to interface name

   Command #1 run: ip a

   e.g.:

```
root@pve101:~# 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 noprefixroute
       valid_lft forever preferred_lft forever
2: nic4: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:04 brd ff:ff:ff:ff:ff:ff
    altname enp4s0
    altname enx005056101004
3: nic1: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq master vmbr0 state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:01 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    altname enx005056101001
4: nic5: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:05 brd ff:ff:ff:ff:ff:ff
    altname enp12s0
    altname enx005056101005
5: nic2: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:02 brd ff:ff:ff:ff:ff:ff
    altname enp19s0
    altname enx005056101002
6: nic6: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:06 brd ff:ff:ff:ff:ff:ff
    altname enp20s0
    altname enx005056101006
7: nic3: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:03 brd ff:ff:ff:ff:ff:ff
    altname enp27s0
    altname enx005056101003
8: vmbr0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default qlen 1000
    link/ether 00:50:56:10:10:01 brd ff:ff:ff:ff:ff:ff
    inet 10.26.20.101/24 scope global vmbr0
       valid_lft forever preferred_lft forever
root@pve101:~# _
```

   Step 2: Compare to Table 10 (if you are training in our lab)

   Step 3: Make sure that the interface we are editing in subsequent steps is one which has access to VLAN 20

5. If you use VLANs and ping was unsuccessful some modifications of this file are required.
   Command #1 run: vi /etc/network/interfaces
   e.g.:

```
root@pve101:~# vi /etc/network/interfaces
```

6. This is the configuration created by the installer without VLAN
   e.g.:

```
auto lo
iface lo inet loopback

iface nic0 inet manual

auto vmbr0
iface vmbr0 inet static
        address 10.26.20.101/24
        gateway 10.26.20.1
        bridge-ports nic0
        bridge-stp off
        bridge-fd 0

iface nic1 inet manual

iface nic2 inet manual

iface nic3 inet manual

iface nic4 inet manual

iface nic5 inet manual


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

8. If you use VLANs (TRUNK ports), you will need to modify your initial config as follows, except substituting your correct values.

   Step 1: Using the arrows, move the cursor to the line: auto vmbr0

   Step 2: Press [End]

   Step 3: Press lowercase [a] to go to insert mode (add)

   Step 4: Add the text: .20 to make the line read: auto vmbr0.20

   Step 5: Use the down arrow to move down to the the line: iface vmbr0 inet static

   NOTE : When you use the up or down arrows, the editor will shift from insert mode to command mode, so you will have to press [a] again on the second line.

   Step 6: Use the arrows to place the cursor over the blank space following: vmbr0

   Step 7: Press lowercase [i] to go to insert mode (at)

   Step 8: Modify the line to read: iface vmbr0.20 inet static

   e.g.:

```
auto lo
iface lo inet loopback

iface nic0 inet manual

auto vmbr0.20
iface vmbr0.20 inet static
        address 10.26.20.101/24
        gateway 10.26.20.1
```

   Step 9: We are using VLAN 20, so change the first two lines with “vmbr0” to: vmbr0.20

   Step 10: Now move the cursor to the last digit of the gateway and press [a] to go to add

   Step 11: Press [ENTER] twice (to move existing entries down and insert a blank line)

   Step 12: Add the lines: ‘auto vmbr0’ and ‘iface vmbr0 inet manual’

   e.g.:

```
auto vmbr0.20
iface vmbr0.20 inet static
        address 10.26.20.101/24
        gateway 10.26.20.1
```

Step 13: Now move to the end of the last line and press [a] followed by [ENTER] to add the following lines

e.g.:

```
        bridge-vlan-aware yes
        bridge-vids 20-26
```

9. When complete, your file should look like this

Step 1: Press [ESC] to leave insert mode

Step 2: Press [:] to enter command mode

Step 3: Enter [w] and [q] to write and quit

e.g.:

```
auto lo
iface lo inet loopback

iface nic1 inet manual

auto vmbr0.20
iface vmbr0.20 inet static
        address 10.26.20.101/24
        gateway 10.26.20.1

auto vmbr0
iface vmbr0 inet manual
        bridge-ports nic1
        bridge-stp off
        bridge-fd 0
        bridge-vlan-aware yes
        bridge-vids 20-26

iface nic4 inet manual

iface nic5 inet manual

iface nic2 inet manual

iface nic6 inet manual

iface nic3 inet manual


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

Command #1 run: ifreload -a

11. Ping the gateway again

    Command #1 run: ping 10.26.20.1

    e.g.:

```
root@pve101:~# ping 10.26.20.1
PING 10.26.20.1 (10.26.20.1) 56(84) bytes of data.
64 bytes from 10.26.20.1: icmp_seq=1 ttl=64 time=0.258 ms
64 bytes from 10.26.20.1: icmp_seq=2 ttl=64 time=0.249 ms
64 bytes from 10.26.20.1: icmp_seq=3 ttl=64 time=0.276 ms
64 bytes from 10.26.20.1: icmp_seq=4 ttl=64 time=0.262 ms
^C
--- 10.26.20.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3106ms
rtt min/avg/max/mdev = 0.249/0.261/0.276/0.009 ms
root@pve101:~# ^C
root@pve101:~# _
```

    NOTE : You will need to press [CTL-C] to stop the ping

SBS LAB – (GUI) Logging on to PVE GUI

Variables for this lab (or other as appropriate for your environment):

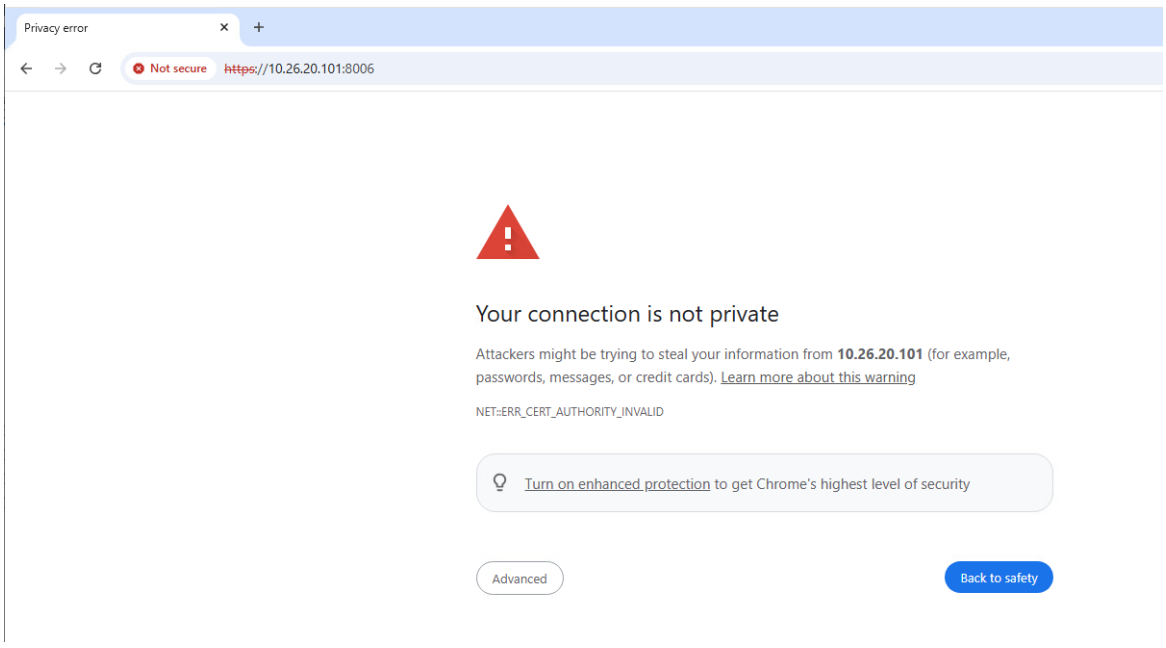
- Web browser
- Username: root
- Password: P@ssw0rd
- URL: https://10.26.20.XYZ:8006

1. Now we access PVE by its GUI interface

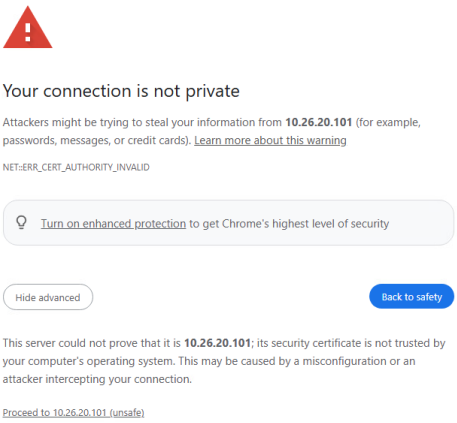
   Step 1: Open a browser to: https://10.26.20.XYZ:8006

   Step 2: Click on: Advanced

   e.g.:



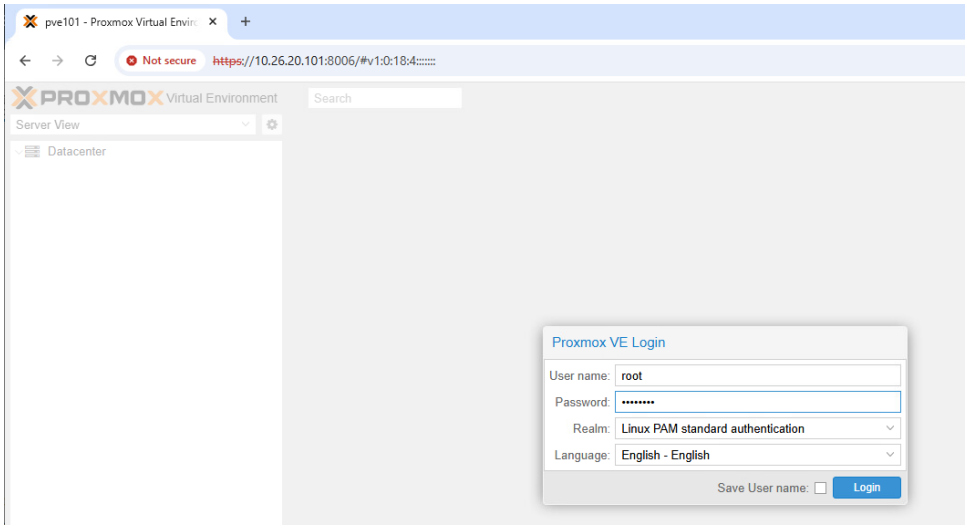
Step 3: Click on: Proceed...



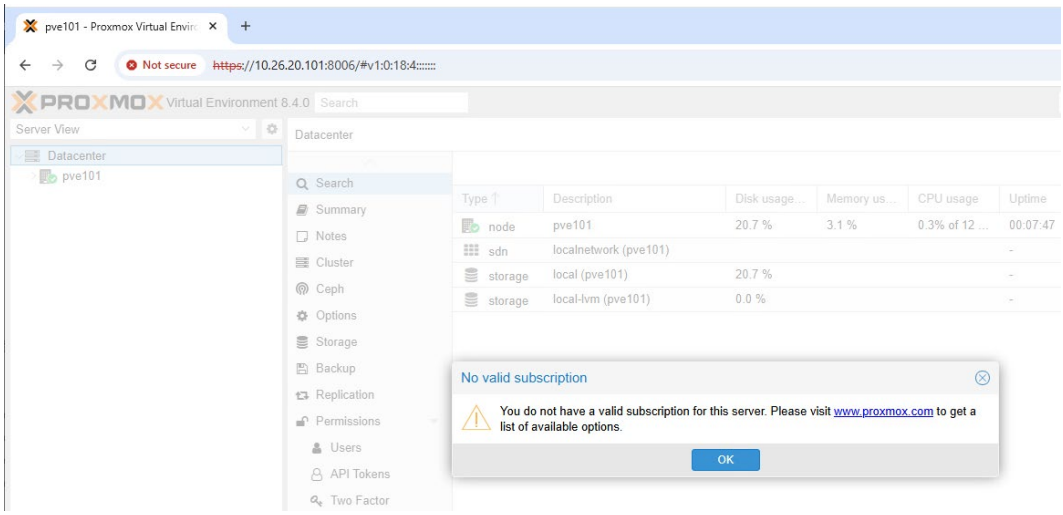
e.g.:

Step 4: Log on with the credential: root and the password you set in the Realm: Linux PAM standard authentication

e.g.:

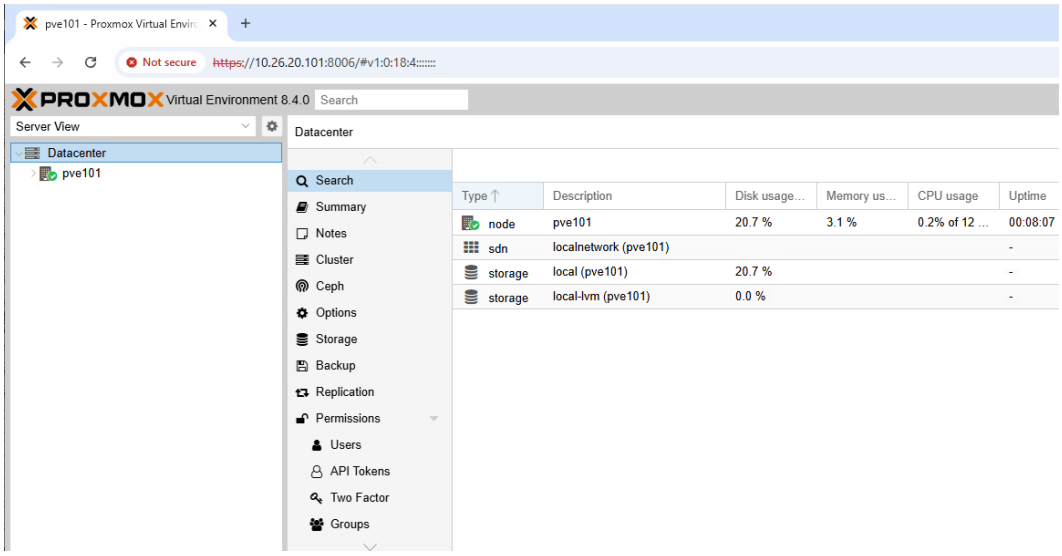


e.g.:



2. This is your basic, unconfigured PVE interface

e.g.:

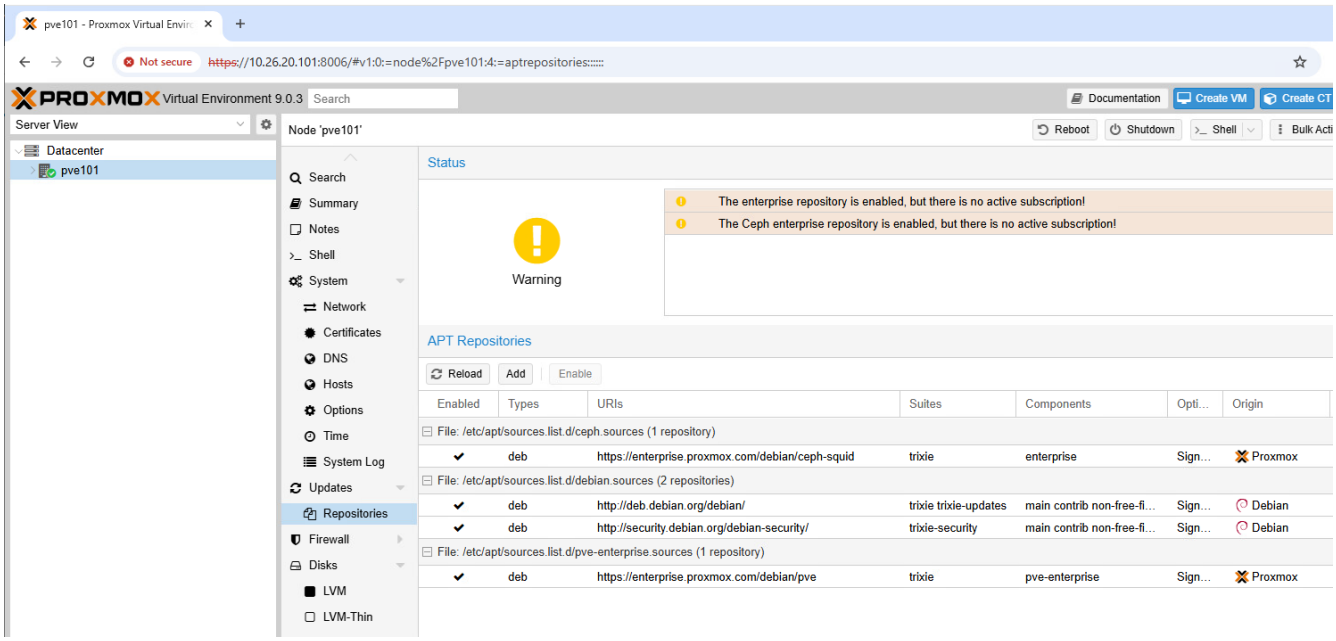


SBS LAB – (GUI) Configuring and Running Updates for PVE

1. Since we're going to be using the free version of PVE for our training lab, we will be disabling the Enterprise repos and adding the pve-no-subscription repos.

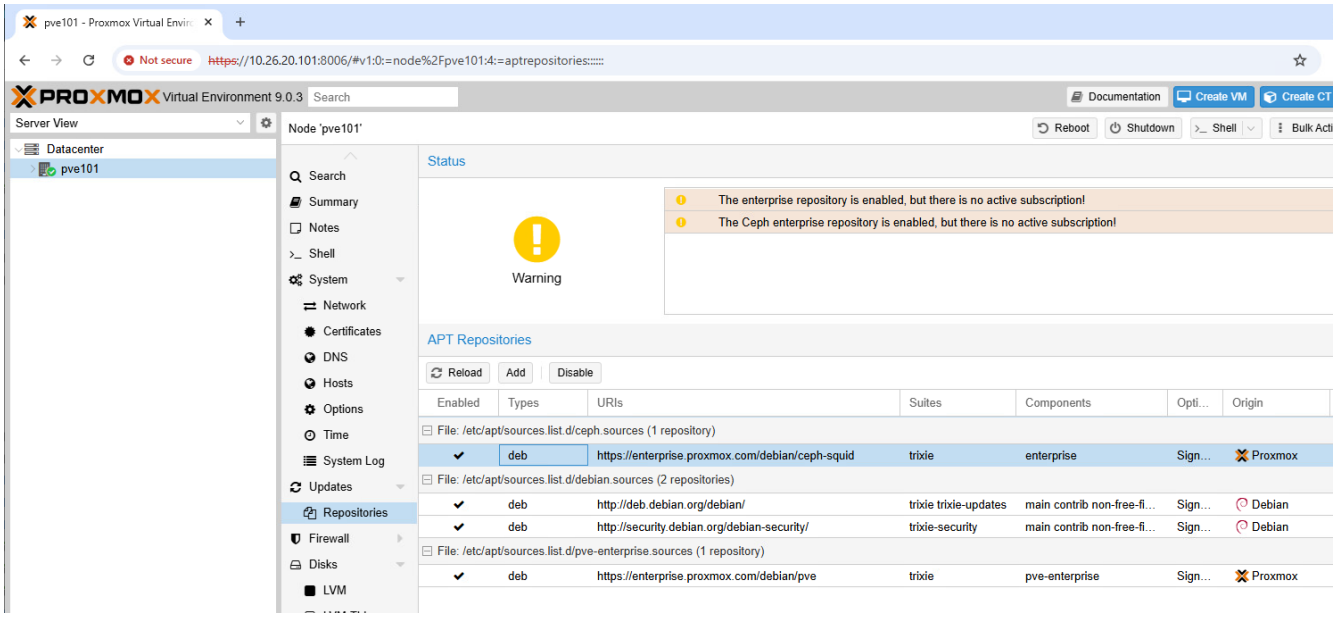
   Step 1: Go to: pveXYZ > Updates > Repositories

   e.g.:



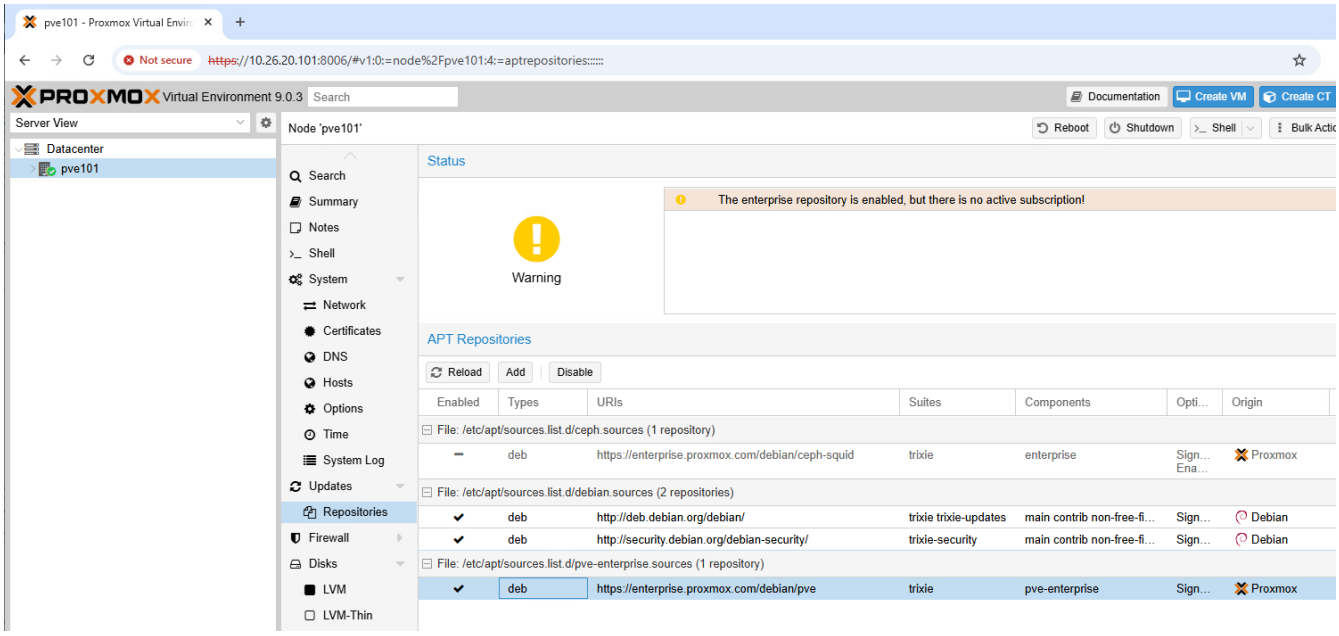
   Step 1: Select the first enterprise repo and click: Disable

   e.g.:



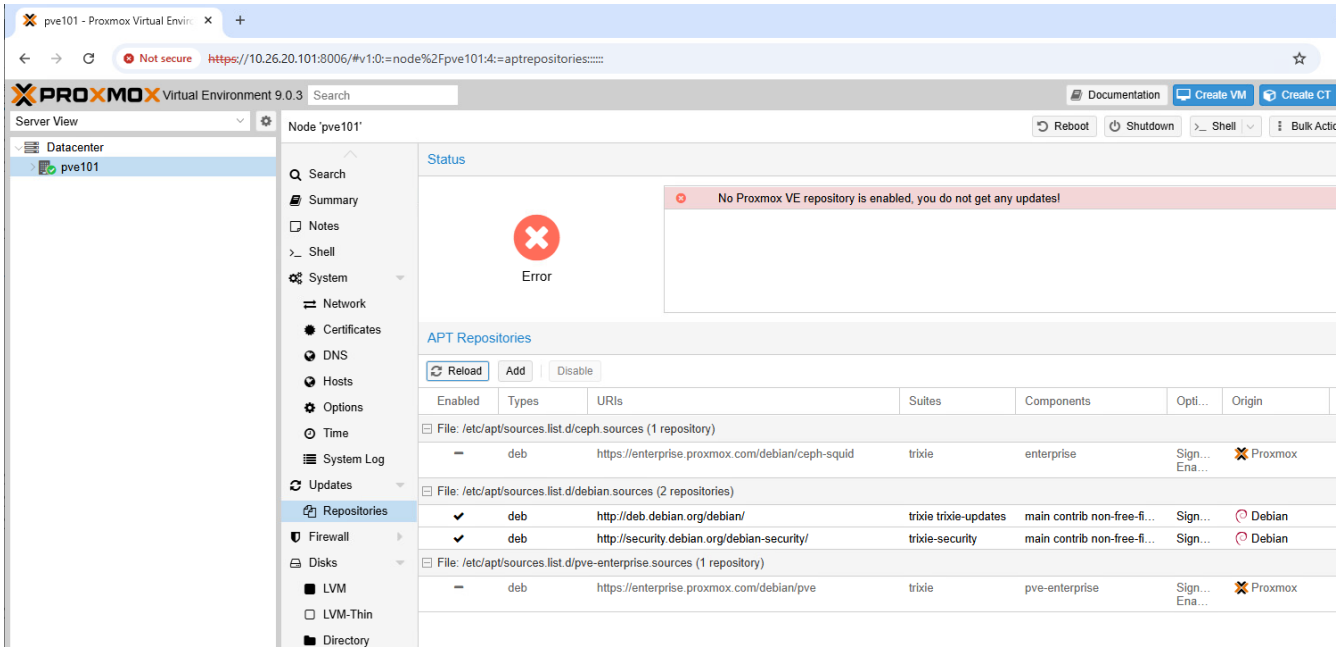
Step 2: Select the second enterprise repo and click: Disable

e.g.:

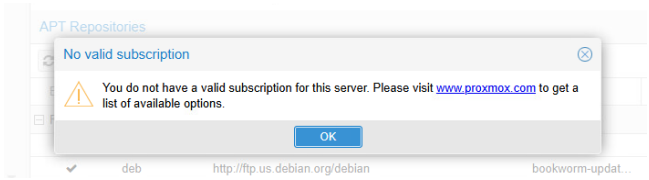


Step 3: Click Reload

e.g.:



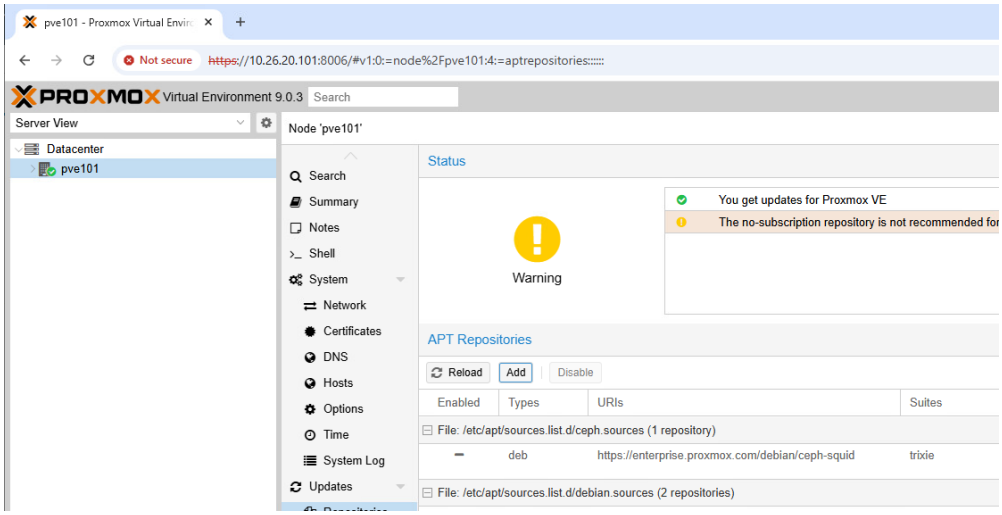
e.g.:



2. Now add the “No-Subscription repos for both PVE and Ceph

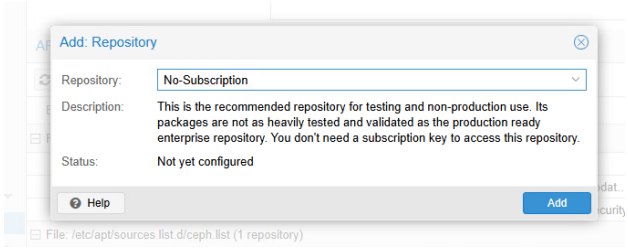
Step 1: Click Add

e.g.:



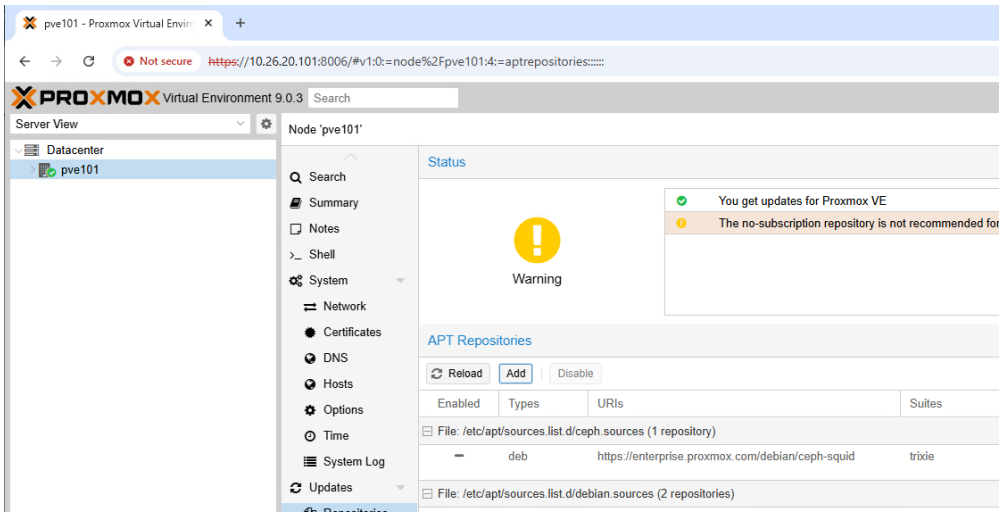
Step 2: Choose “No-Subscription” repo

e.g.:



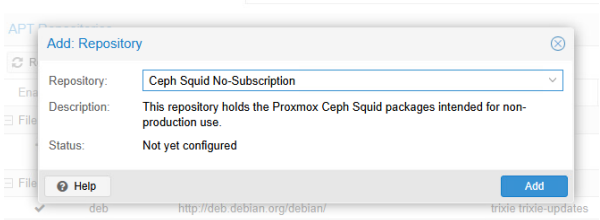
Step 3: Click Add

e.g.:



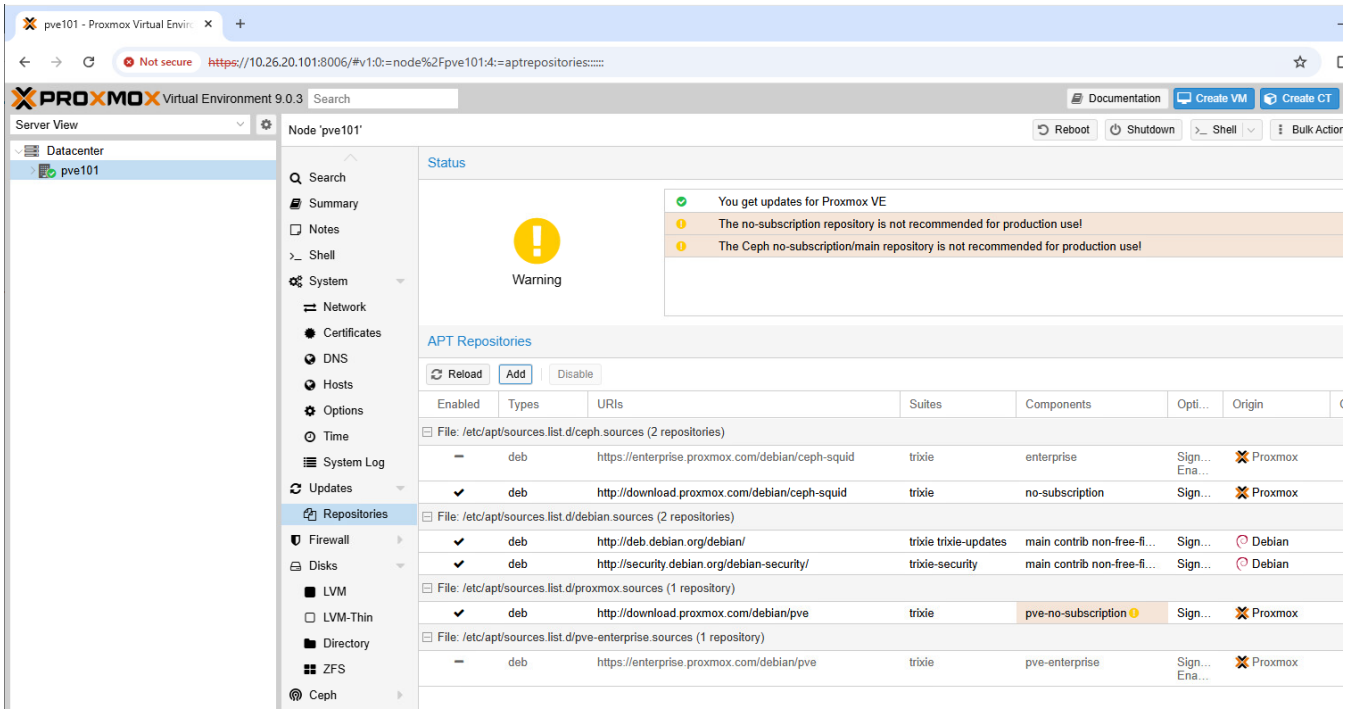
Step 4: Choose the “Ceph Squid No-Subscription” repo

e.g.:



Step 5: Click Reload

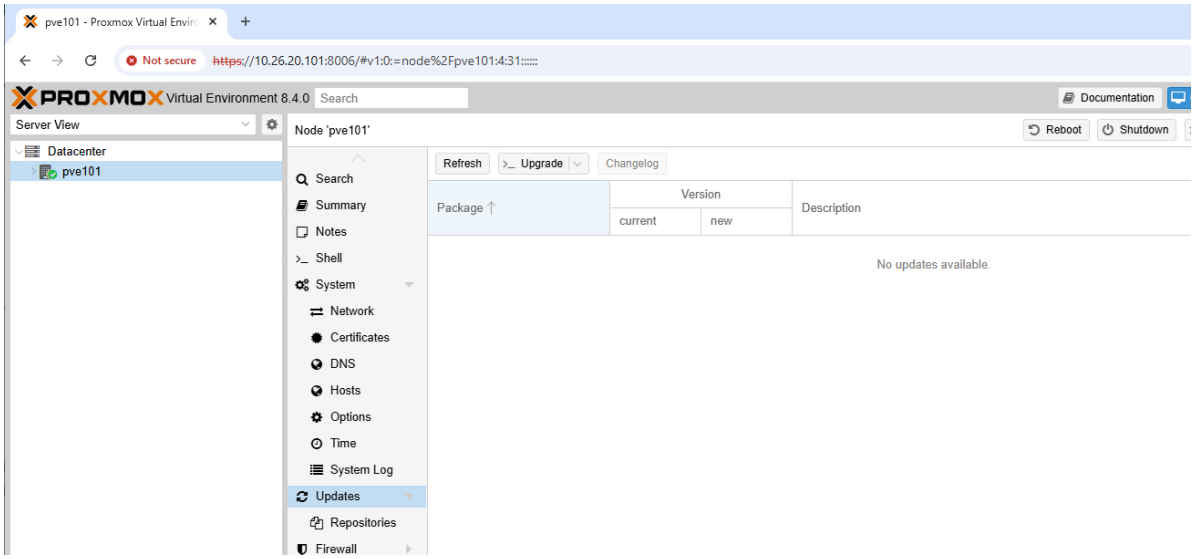
e.g.:



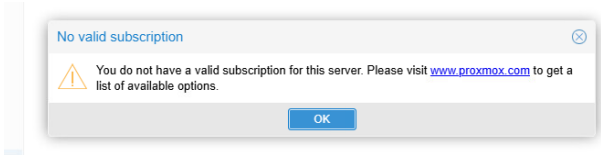
Step 6: Go to: pveXYZ > Updates

Step 7: Click Refresh

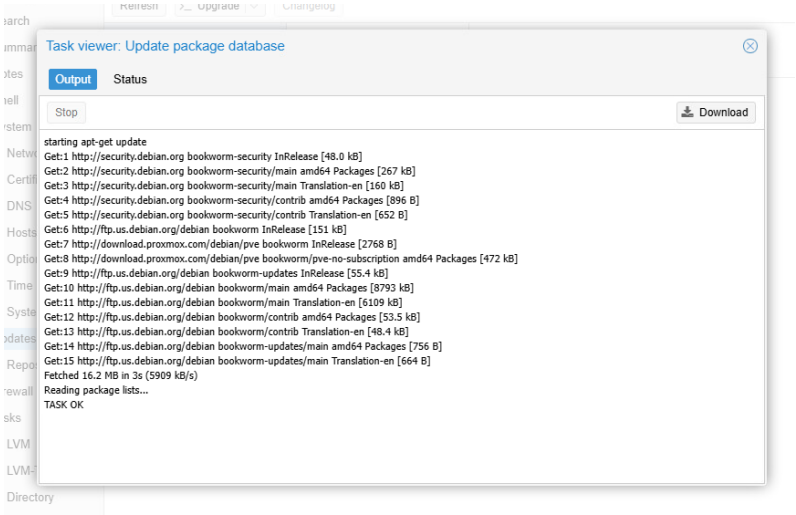
e.g.:



e.g.:

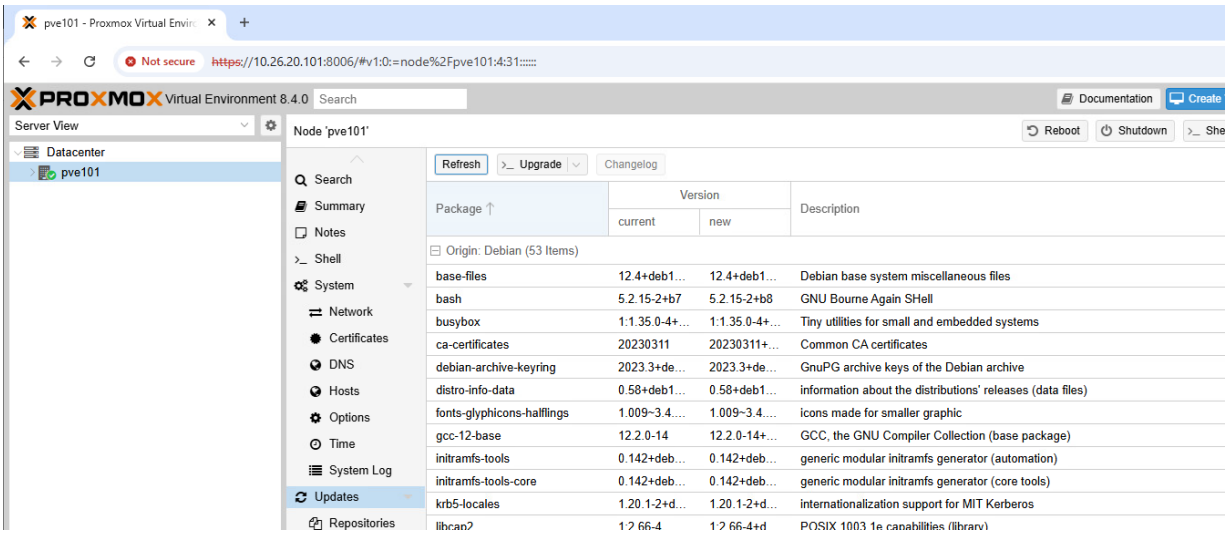


e.g.:



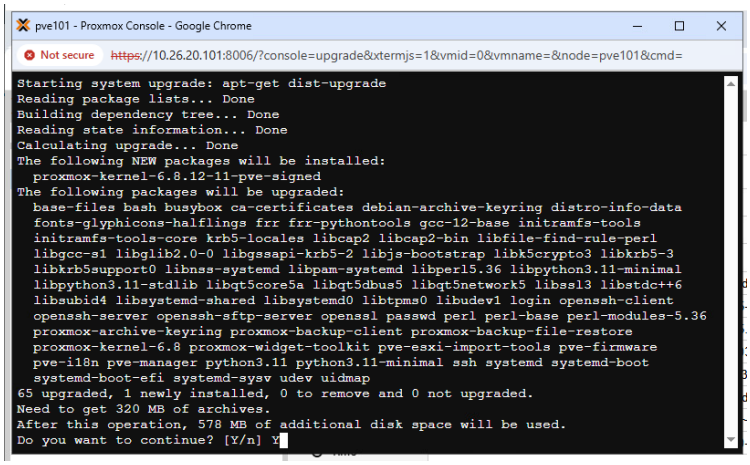
e.g.:

Step 8: Click: Upgrade



Step 9: Enter [Y]

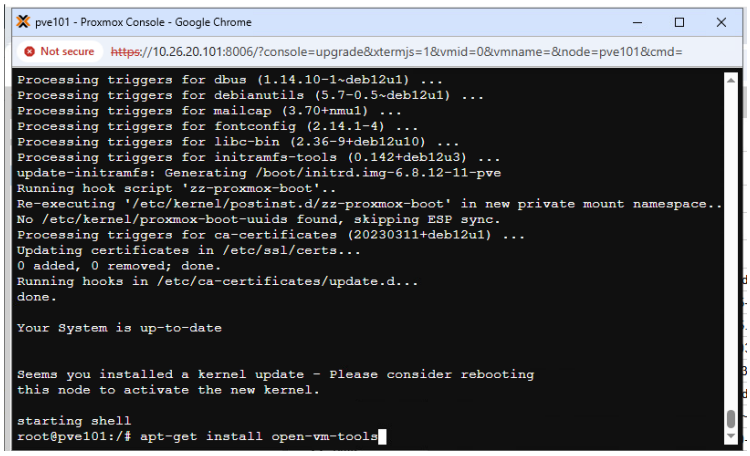
e.g.:



Step 10: **Only if you are running a nested PVE lab**

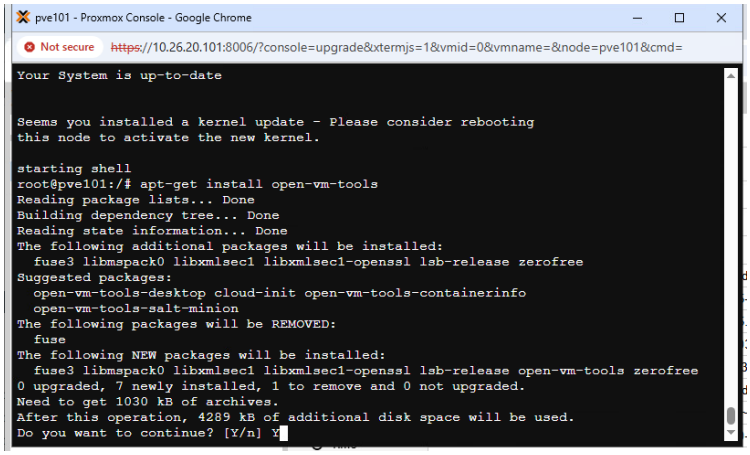
Command #1 run: apt install open-vm-tools

e.g.:



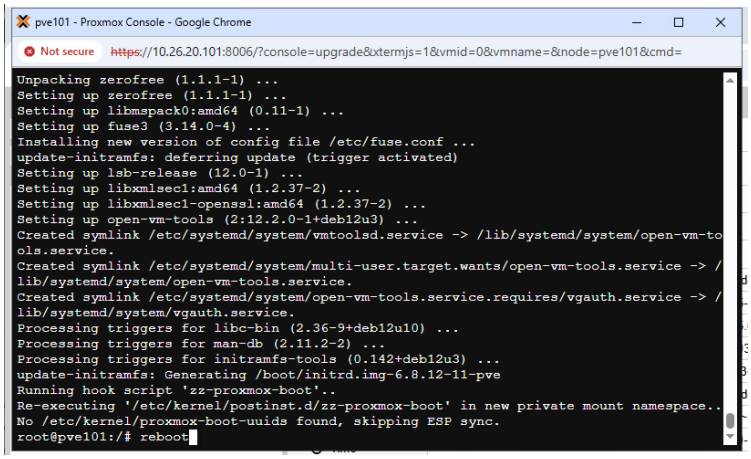
Step 11: Enter [Y]

e.g.:



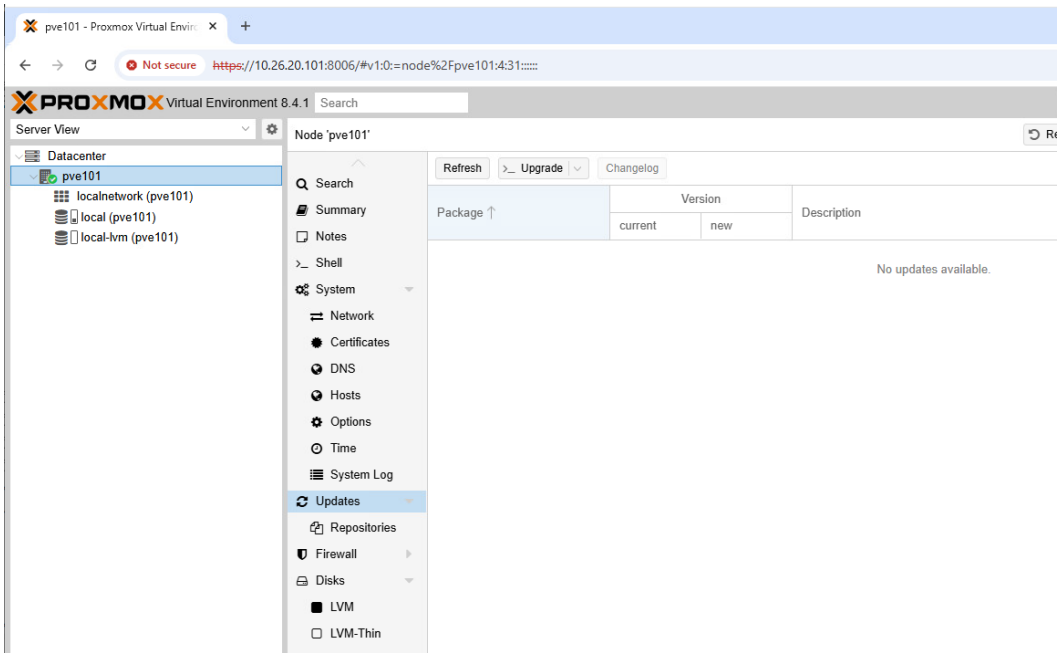
Step 12: Reboot is not required/recommended unless a new kernel version was installed

e.g.:



Step 13: Wait for the system to reboot and refresh the browser

e.g.:



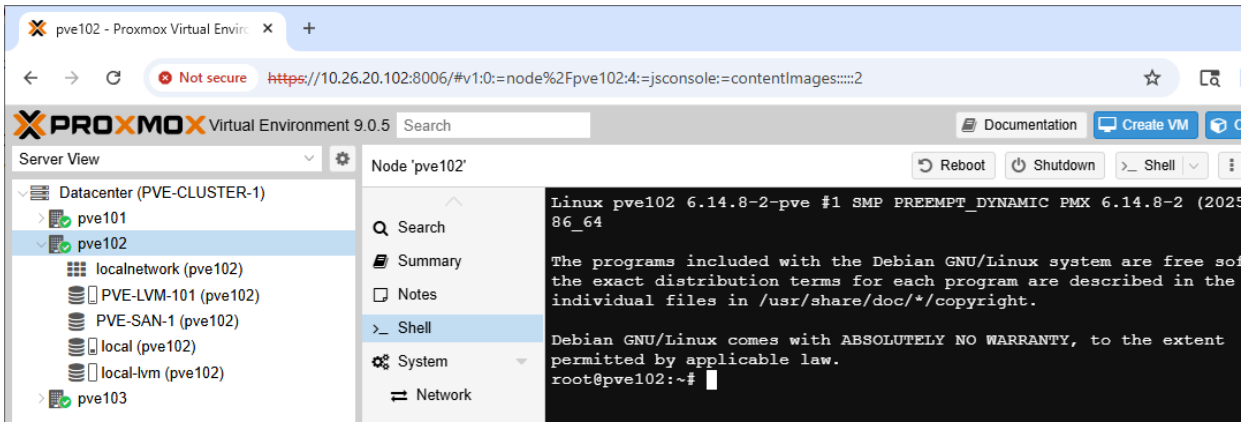
SBS LAB –(CLI) Time Config for PVE

Although PVE is configured by default to get time from debian.pool.ntp.org, it is a good idea to configure the time sources manually. This is especially important if there is no outside network access or you prefer (or are required) to use a specific NTP server(s).

1. Edit the basic configuration

   Step 1: Open a shell on your PVE node

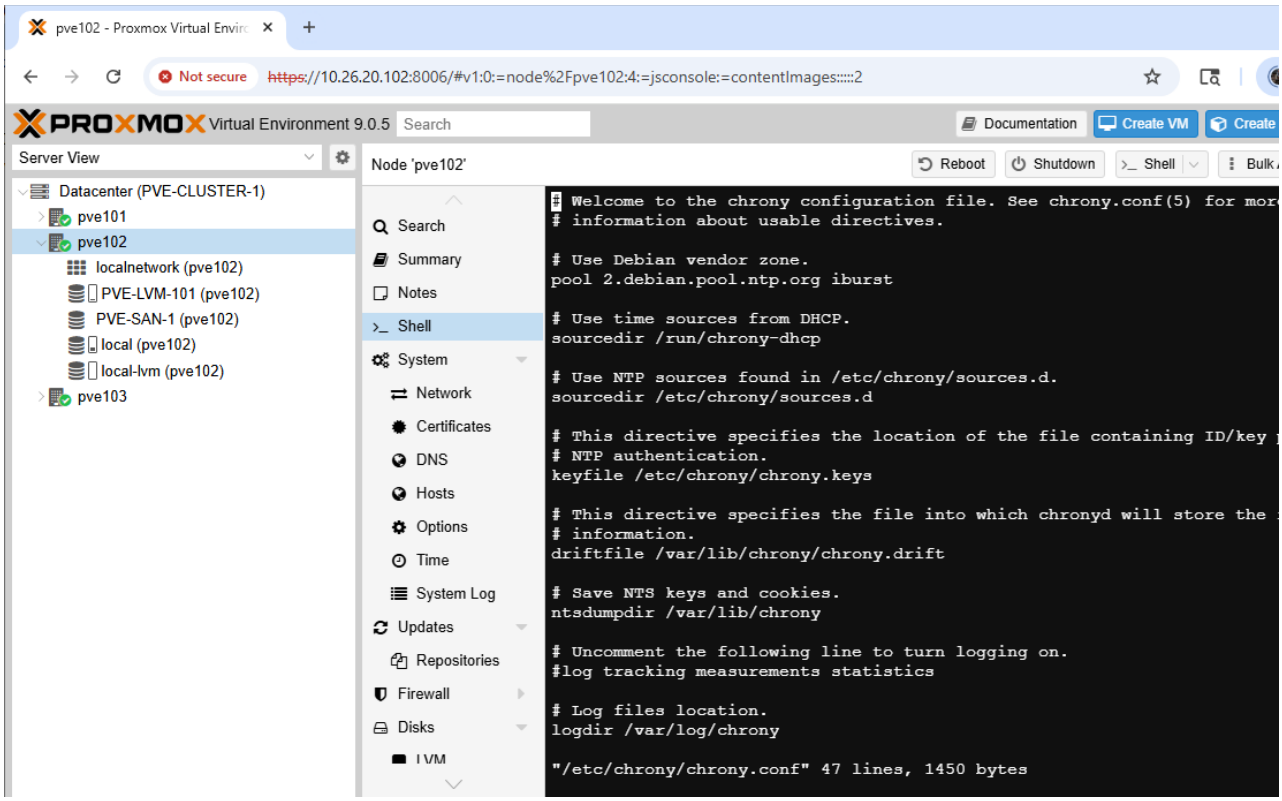
   e.g.:



   Step 2: Edit chrony.conf

   Command #1 run: vi /etc/chrony/chrony.conf

   e.g.:



Step 3: Comment out unneeded or unwanted properties

e.g.:

```
# Welcome to the chrony configuration file. See chron
# information about usable directives.

# Use Debian vendor zone.
#pool 2.debian.pool.ntp.org iburst

# Use time sources from DHCP.
#sourcedir /run/chrony-dhcp

# Use NTP sources found in /etc/chrony/sources.d.
sourcedir /etc/chrony/sources.d
```

NOTE : We commented out the Debian vendor zone and DHCP time sources

Step 4: Write and quit

e.g.:

```
# Uncomment the following line to turn
#log tracking measurements statistics

# Log files location.
logdir /var/log/chrony

:wq
```

2. Now add additional sources to a file in the folder: /etc/chrony/sources.d/

Step 1: To add pool NTP servers

Command #1 run: echo 'pool 0.us.pool.ntp.org iburst' >> /etc/chrony/sources.d/ntp-pool.sources

e.g.:

```
Linux pve102 6.14.8-2-pve #1 SMP PREEMPT_DYNAMIC PMX 6.14.8-2 (2025-07-22T10:04Z) x
86_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.
root@pve102:~# vi /etc/chrony/chrony.conf
root@pve102:~# echo 'pool 0.us.pool.ntp.org iburst' >> /etc/chrony/sources.d/ntp-po
ol.sources
```

NOTE : Repeat the last command for as many pools as you would like to add

Step 2: If you wish to add a time server directly (in addition to or in place of pools)

Command #1 run: echo 'server 129.6.15.28 iburst' >> /etc/chrony/sources.d/ntp-server.sources

e.g.:

```
root@pve102:~# echo 'server 129.6.15.28 iburst' >> /etc/chrony/sources.d/ntp-server
.sources
```

NOTE : Repeat the last command for as many time servers as you would like to add

Step 3: Restart Chrony

Command #1 run: systemctl restart chronyd

e.g.:

```
root@pve102:~# systemctl restart chronyd
root@pve102:~#
```

Step 4: Now verify your NTP timeservers.

Command #1 run: chronyc -n sources

Command #2 run: chronyc -n tracking

e.g.:

```
root@pve102:~# echo 'server 129.6.15.28 iburst' >> /etc/chrony/sources.d/ntp-serv
.sources
root@pve102:~# systemctl restart chronyd
root@pve102:~# chronyc -n sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^+ 50.205.57.38                  1   6    17    18   -540us[ -361us] +/-   34ms
^- 74.208.25.46                  3   6    17    17  +7438us[+7438us] +/-   77ms
^+ 141.11.228.173                2   6    17    17   +247us[ +247us] +/-   34ms
^* 23.168.24.210                 2   6    17    17  +1195us[+1374us] +/-   31ms
^- 129.6.15.28                   1   6    17    17   -724us[ -724us] +/-   37ms
root@pve102:~# chronyc -n tracking
Reference ID    : 17A818D2 (23.168.24.210)
Stratum         : 3
Ref time (UTC)  : Fri Aug 15 18:29:41 2025
System time     : 0.000000173 seconds slow of NTP time
Last offset     : +0.000179015 seconds
RMS offset      : 0.000179015 seconds
Frequency       : 13.689 ppm slow
Residual freq   : +14.965 ppm
Skew            : 0.188 ppm
Root delay      : 0.059414648 seconds
Root dispersion : 0.002412317 seconds
Update interval : 1.3 seconds
Leap status     : Normal
root@pve102:~#
```

Certificates for PVE

Having a valid SSL Certificate configured for environments such as PVE is a must for any Organization to ensure security and meet auditing requirements.

In reality, the self-signed certificate created when you install PVE is valid, it's just that your browser (and other browsers in your organization) don't know it and will always produce a warning. Moreover, the self-signed certificate is generic to PVE and not to your Organization.

There are many tutorials and videos online about using Let's Encrypt certificates with PVE. Let's Encrypt is one of the greatest things since sliced bread, and there is nothing wrong with using it for public-facing enterprise websites. Using Let's Encrypt with PVE is ridiculously easy except for the fact that we probably don't want PVE to be public-facing! You also can't use Let's Encrypt for any domains where you don't (or can't) control the public DNS for that TLD. Think: acme.local and any other .local domain.

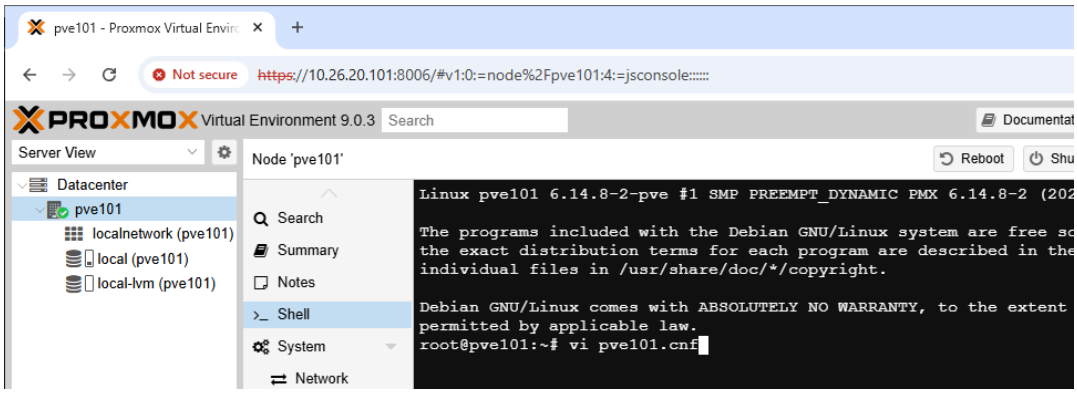
The last consideration is that for Let's Encrypt to work, you need to NAT port 80 from a public IP to each PVE node and register each node in public DNS; nobody wants that in an Enterprise environment. While there are workarounds for Let's Encrypt to function within private networks, the certificate will always be public.

We are going to create certificates for PVE within our environment and apply them on our Domain CA, just like you should in your Organization.

SBS LAB – (GUI) Certificates for PVE

IN PRODUCTION: Because the first node in a cluster will overwrite the certificate settings of all subsequent nodes that join the cluster, you would perform the following only on the first node.

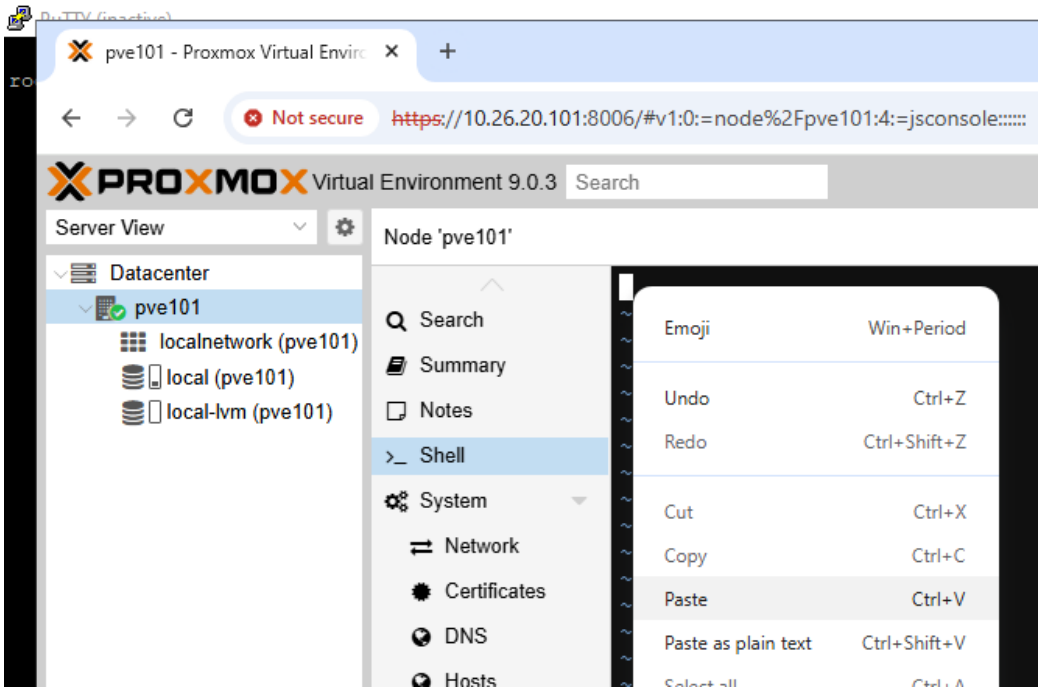
1. First we must create a CNF file for our certificate so that we can include Subject Alternative Names (SAN), which are what modern browsers use to identify our site.
   Step 1: Open a shell
   Command #1 run: vi pve101.cnf
   e.g.:



   Step 2: Paste the following contents into the file by pressing [i], the right-click to select [paste], then replace XYZ with your number

```
[ req ]
  default_bits = 2048
  prompt = no
  default_md = sha256
  req_extensions = req_ext
  distinguished_name = dn

  [ dn ]
  CN = pveXYZ.lab.vmsources.com

  [ req_ext ]
  subjectAltName = @alt_names

  [ alt_names ]
  DNS.1 = pveXYZ.lab.vmsources.com
```

e.g.:



e.g.:

```
[ req ]
    default_bits = 2048
    prompt = no
    default_md = sha256
    req_extensions = req_ext
    distinguished_name = dn

    [ dn ]
    CN = pve101.lab.vmsources.com

    [ req_ext ]
    subjectAltName = @alt_names

    [ alt_names ]
    DNS.1 = pve101.lab.vmsources.com
~
~
```

Step 3: Press [ESC] and then write and quit with :wq

e.g.:

```
~
~
~
~
:wq
```

Step 4: Create a private key

Command #1 run: openssl genrsa -out /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key 2048

e.g.:

```
permitted by applicable law.
root@pve101:~# vi pve101.cnf
root@pve101:~# openssl genrsa -out /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key 2048
```

Step 5: Now create your Certificate Signing Request (CSR) with the following command

Command #1 run: openssl req -new -key /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key -out $HOSTNAME.csr -config $HOSTNAME.cnf

e.g.:

```
permitted by applicable law.
root@pve101:~# vi pve101.cnf
root@pve101:~# openssl genrsa -out /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key 2048
root@pve101:~# openssl req -new -key /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key -out $HOSTNAME.csr -c
onfig $HOSTNAME.cnf
```

Command #1 run: ls

e.g.:

```
root@pve101:~# ls
pve101.cnf  pve101.csr
root@pve101:~#
```

Step 6: View the CSR

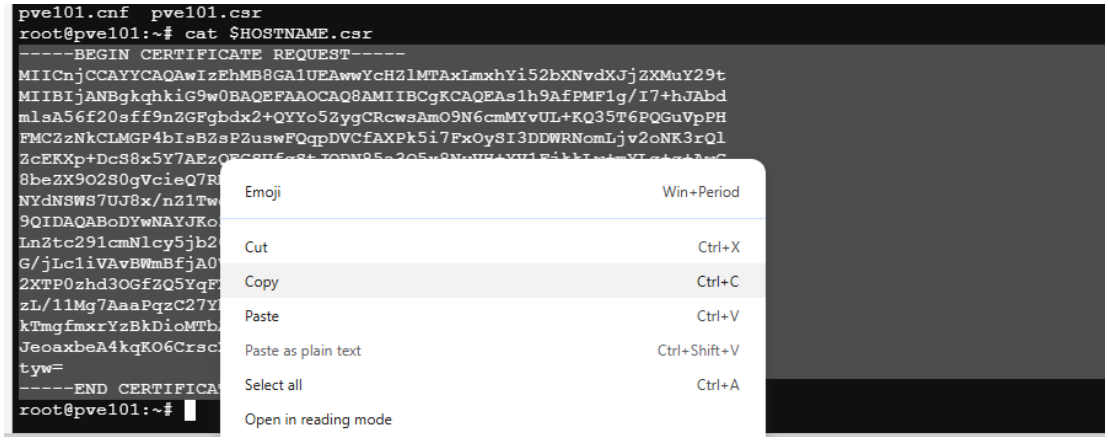
Command #1 run: cat $HOSTNAME.csr

e.g.:

```
root@pve101:~# cat $HOSTNAME.csr
-----BEGIN CERTIFICATE REQUEST-----
MIICnjCCAYYCAQAwIzEhMB8GA1UEAwwYcHZlMTAxLmxhYi52bXNvdXJjZXMuY29t
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAs1h9AfPMF1g/I7+hJAbd
mlsA56f20sff9nZGFgbdx2+QYYo5ZygCRcwsAmO9N6cmMYvUL+KQ35T6PQGuVpPH
FMCZzNkCLMGP4bIsBZsPZuswFQqpDVCfAXPk5i7FxOySI3DDWRNomLjv2oNK3rQl
ZcEKXp+Dc88x5Y7AEzQEGSUfqStJODN85a3Q5x8NyVH+YV1FjkkLw+mYLg+g+AwG
8beZX9O2S0gVcieQ7RMDdxmqIEKiZS/XWHF00UJzwOUcU5eRj70a1AQ2szppHd9z
NYdNSWS7UJ8x/nZ1Two+pcQrtVtjlqkxYqryxMixIeJwk9w+STTQ2isM5LWKxxb5
9QIDAQABoDYwNAYJKoZIhvcNAQkOMScwJTAjBgNVHREEHDAaghhwdmUxMDEubGFi
LnZtc291cmNlcy5jb20wDQYJKoZIhvcNAQELBQADggEBALHIE/CYaFTkcO6PdldS
G/jLc1iVAvBWmBfjA0VNhkKjHQVCSlXy203/rjT8sq7lVjGqFx4mBWbyiJ7c3UlU
2XTP0zhd3OGfZQ5YqFXtNIewc7Xz0PTQoT0qs+gFkZnC0DBGDpc6PR4DNE0wfXMf
zL/11Mg7AaaPqzC27Ybsia/aSd8EXFUQXiRgK8zcoqiXL+/uOHhjCAT2kIhFstRU
kTmgfmxrYzBkDioMTbA3TYgLYoI4Va/vAJK2JtN1Xdbr29SksJHcNejlRAAKB0Zg
JeoaxbeA4kqKO6CrscXAQIPiHI70HLjCJDeqGIVxlUym89cwddRuLMmZ/+h/C+d3
tyw=
-----END CERTIFICATE REQUEST-----
root@pve101:~#
```

Step 7: Copy the CSR to the clipboard (or paste it in Notepad++ for your records)

e.g.:

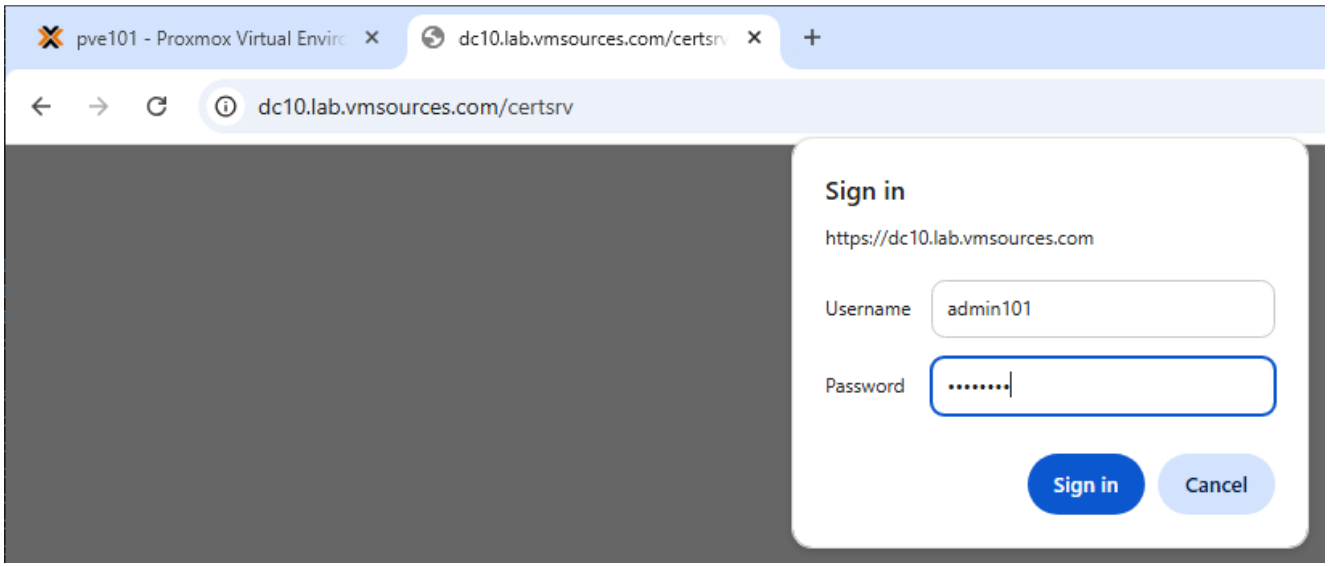


2. Now we request a certificate from our CA

Step 1: In a new tab, browse to: https://dc10.lab.vmsources.com/certsrv

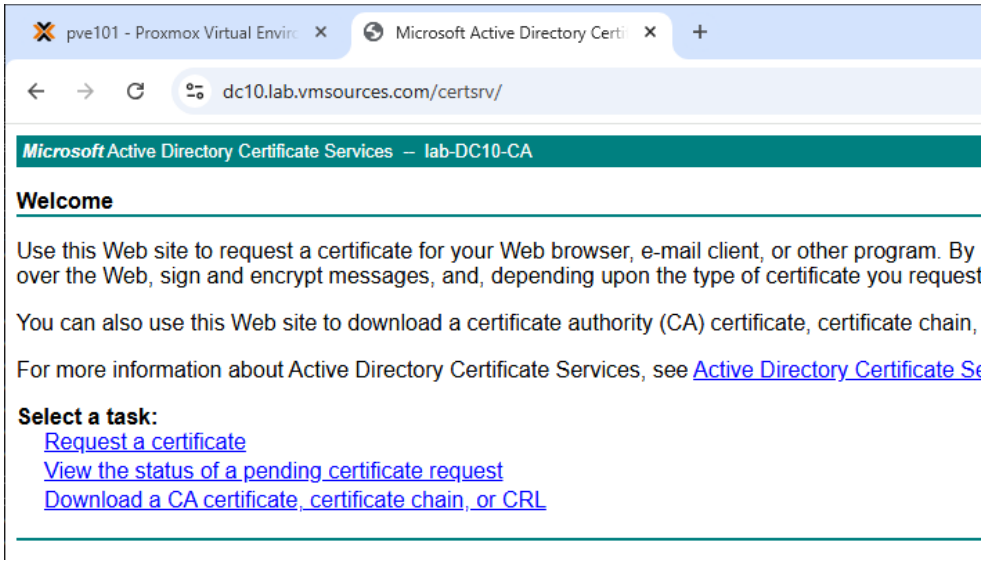
Step 2: Login with your AD credential: lab\adminXYZ

e.g.:



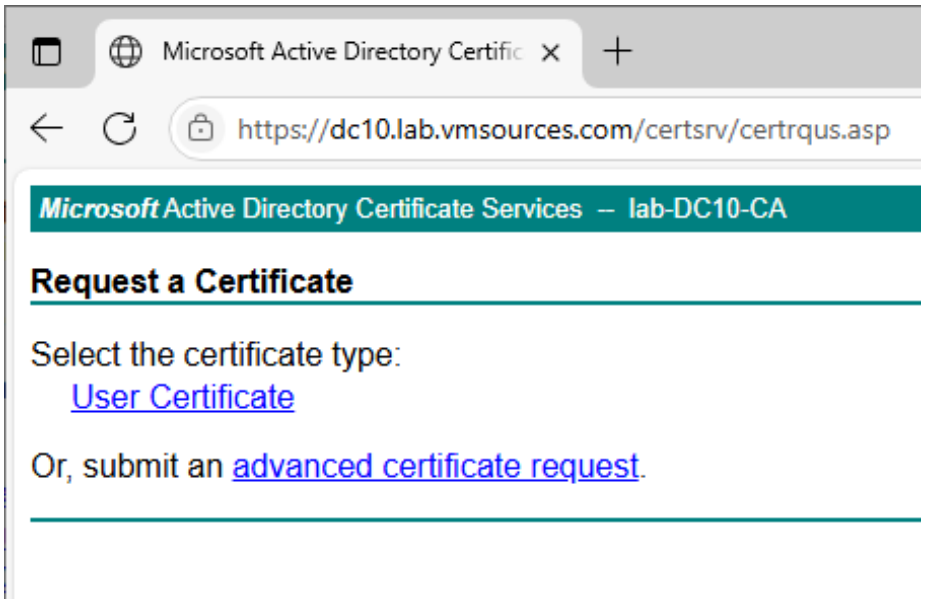
Step 3: Request a certificate

e.g.:



Step 4: Advanced Certificate Request

e.g.:

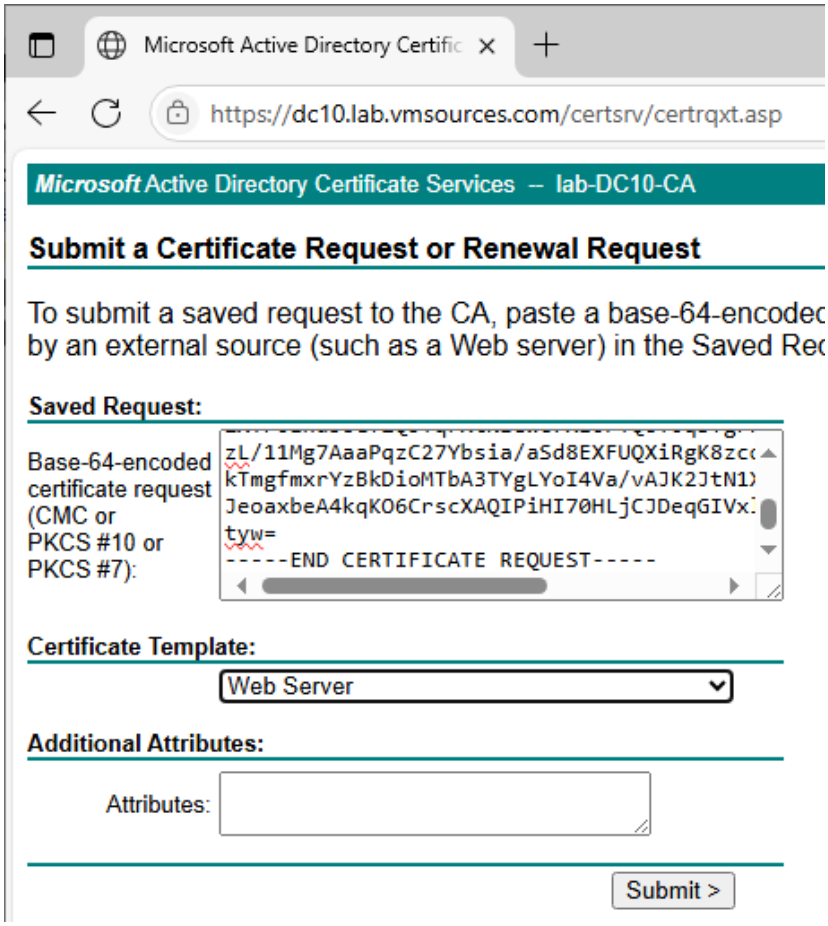


Step 5: Paste your CSR into: Saved Request

Step 6: Certificate Template: Web Server

Step 7: Submit

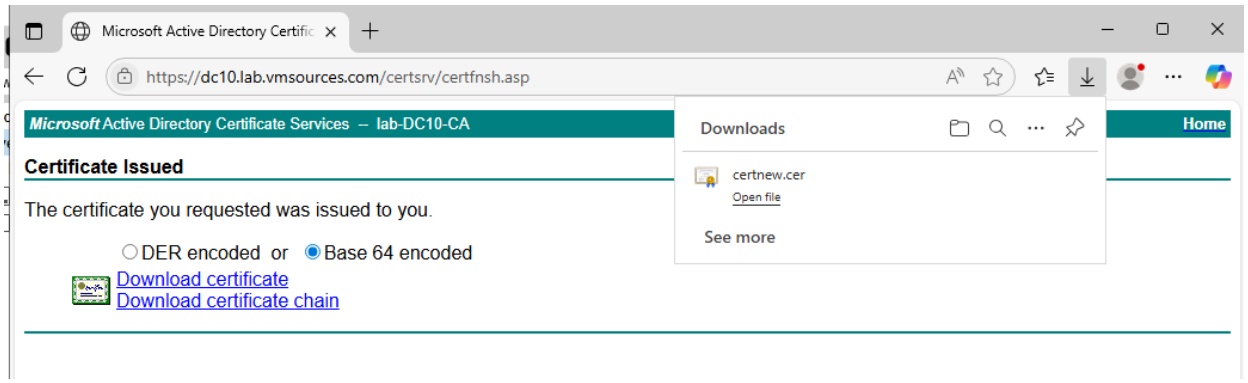
e.g.:



Step 8: Choose: Base 64 encoded

Step 9: Download certificate

e.g.:

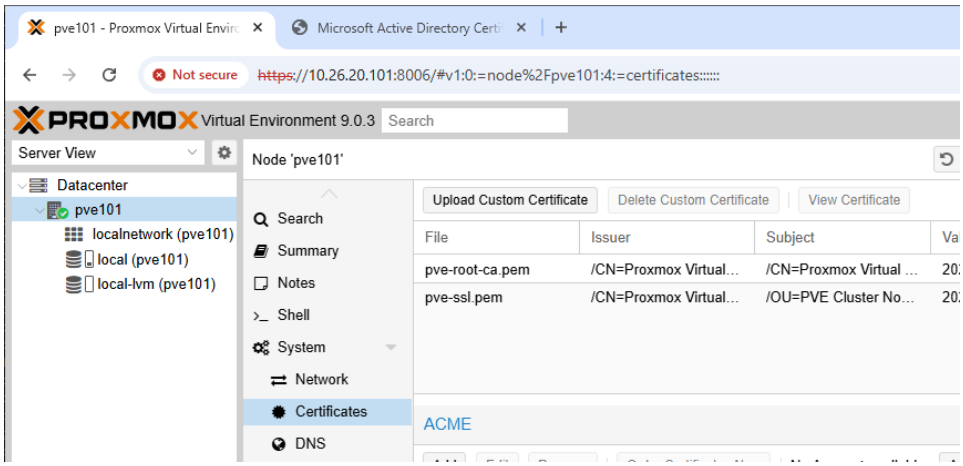


NOTE : Why don't we want the certificate chain: #1: Because the Domain (Windows) CA is already a Truster Root CA on any domain-joined entity. #2: Because we created the Private Key in the correct path on your PVE node.

3. Now we upload the certificate to your PVE node

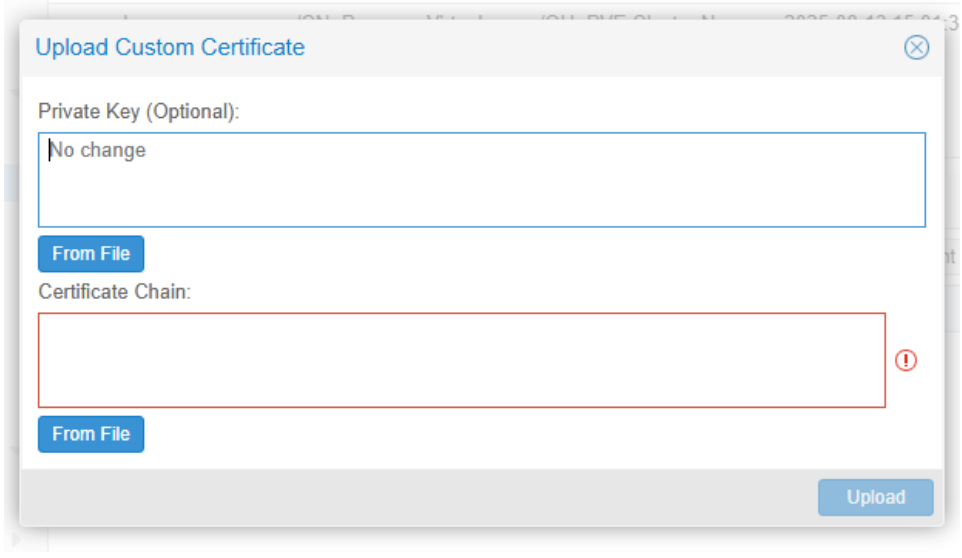
Step 1: pveXYZ > Certificates > Upload Custom Certificate

e.g.:



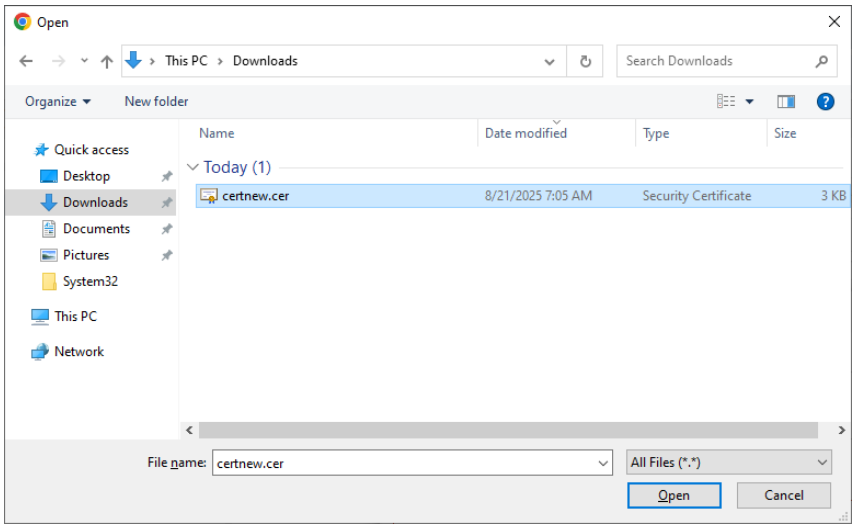
Step 2: Certificate Chain > From File

e.g.:



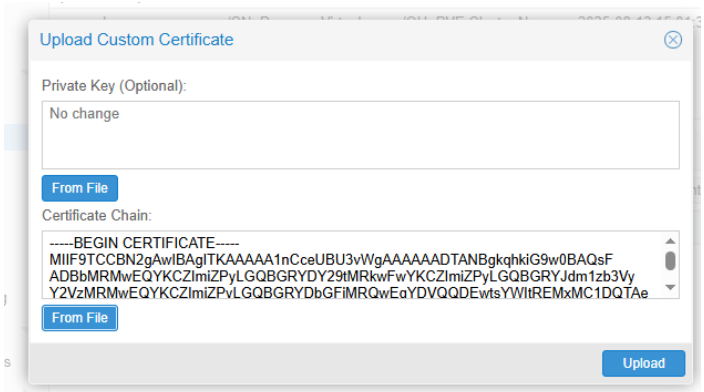
Step 3: Locate your certificate in Downloads > Open

e.g.:

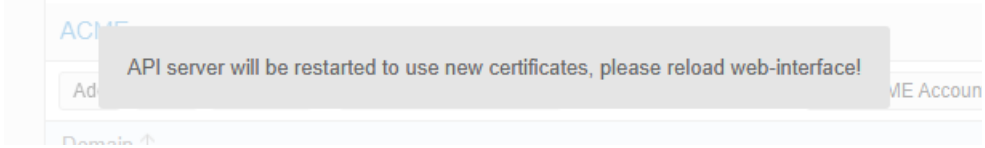


Step 4: Upload

e.g.:

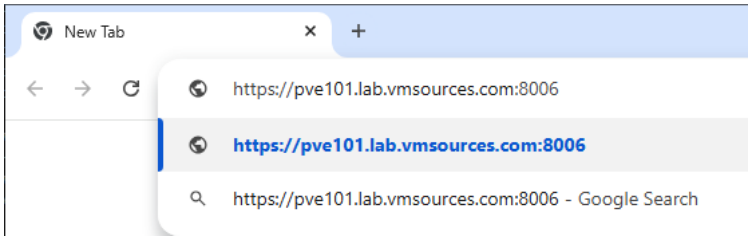


e.g.:

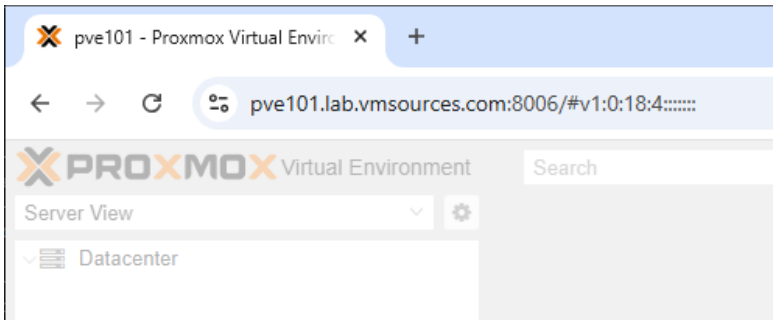


Step 5: Now we need to close and re-open our browser (or use another browser) and access our PVE node using the FQDN and not the IP address.

e.g.:



e.g.:



4. Certificate is working!

1. If you get into trouble:

   Step 1: Delete the following files and regenerate the self-signed certificate,:

   Command #1 run: vi reset_certs.sh

   Step 2: Paste the following script into a file by pressing [i]

```
#!/bin/bash

rm /etc/pve/pve-root-ca.pem
rm /etc/pve/priv/pve-root-ca.key
rm /etc/pve/nodes/$HOSTNAME/pve-ssl.pem
rm /etc/pve/nodes/$HOSTNAME/pve-ssl.key
rm /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.key
rm /etc/pve/nodes/$HOSTNAME/pveproxy-ssl.pem
```

   Command #1 run: chmod u+x reset_certs.sh

   Command #2 run: ./reset_certs.sh

   Command #3 run: pvecm updatecerts -f

   Command #4 run: systemctl restart pveproxy

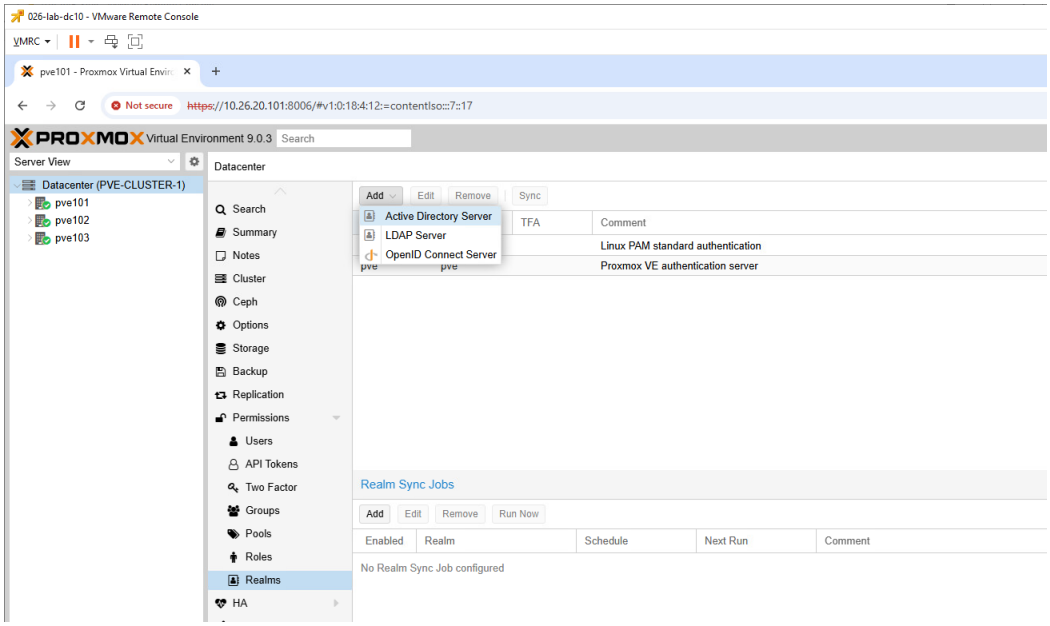
SBS LAB – (GUI) Active Directory Authentication for PVE

Most Enterprise environments are going to require some form of directory based authentication. You should not persistently use 'root' or some other environment-specific login to access PVE, particularly if multiple users are accessing the system.

IN PRODUCTION: Because the first node in a cluster will overwrite the Directory Authentication settings of all subsequent nodes that join the cluster, you would perform the following only on the first node.

Step 1: Datacenter > Permissions > Realms > Add > Active Directory Server

e.g.:



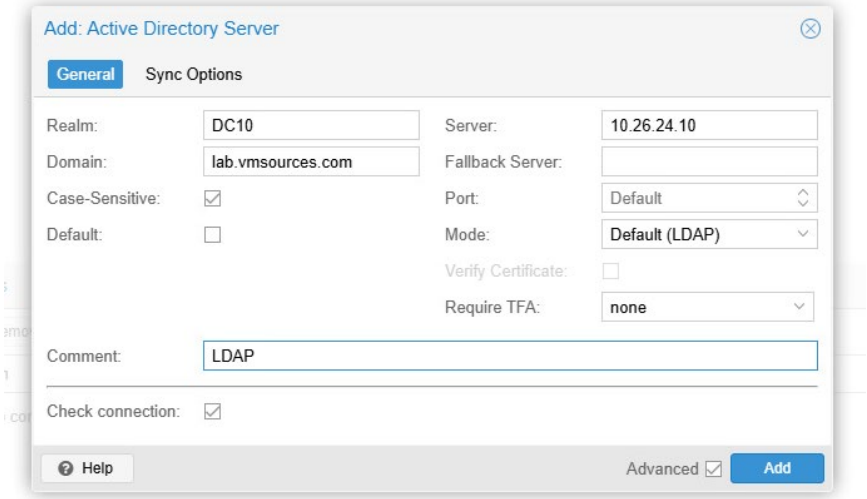
Step 2: Realm: DC10

Step 3: Domain: lab.vmsources.com

Step 4: Server: 10.26.24.10

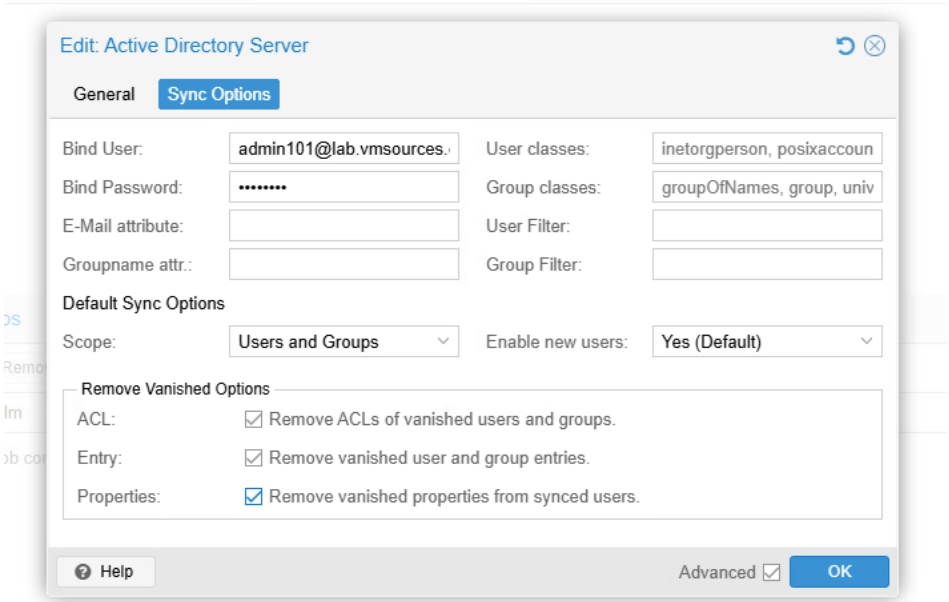
Step 5: Mode: LDAP

Step 6: Comment: LDAP



e.g.:

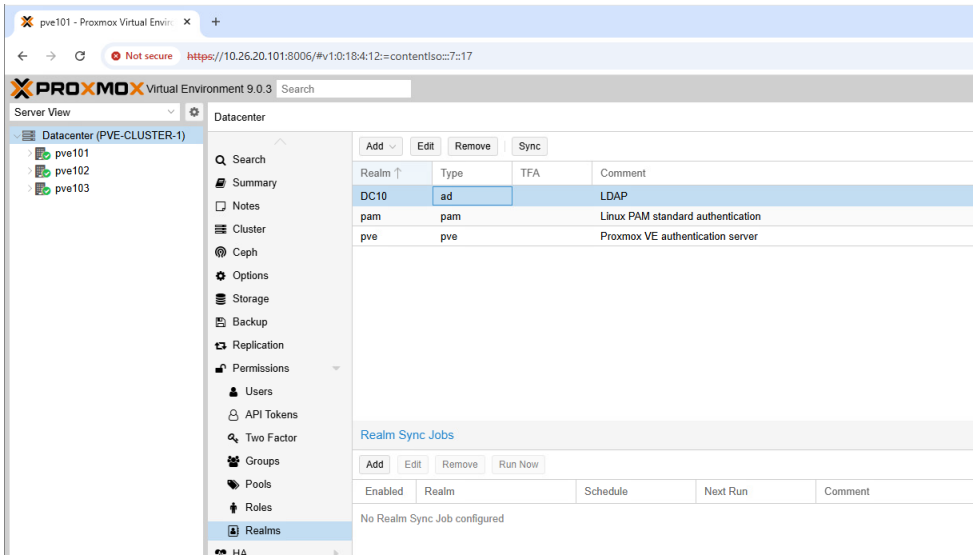
Step 7: Click Sync Options
Step 8: Bind User: adminXYZ@lab.vmsources.com
Step 9: Password: P@ssw0rd
Step 10: Scope: Users and Groups
Step 11: Enable new users: Yes
Step 12: Remove Vanished Options: Check all



e.g.:

Step 13: Sync

e.g.:

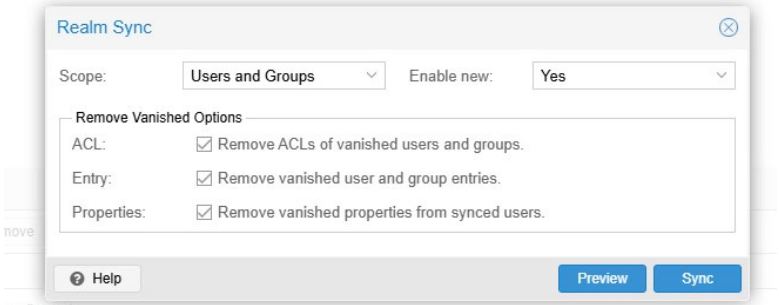


Step 14: Scope: Users and Groups

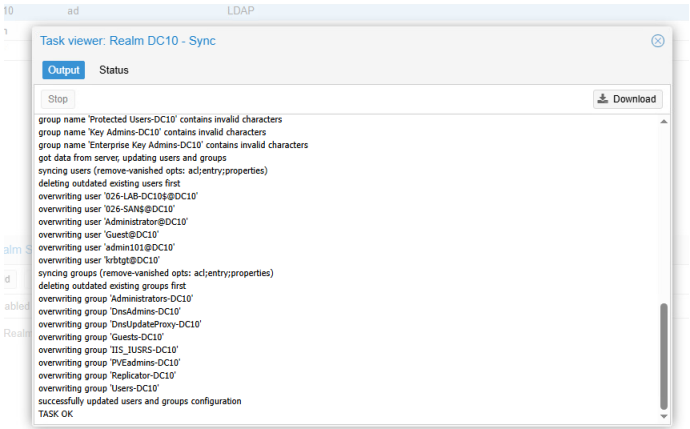
Step 15: Enable new: Yes

Step 16: Remove Vanished Options: Check all

e.g.:



Step 17:

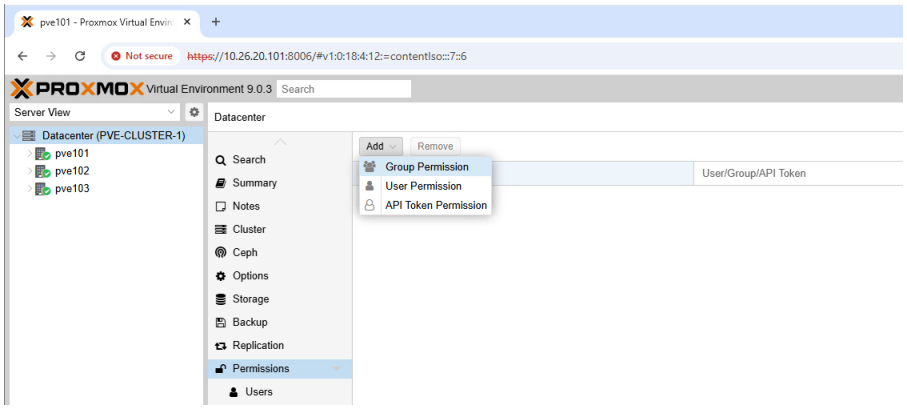


SBS LAB – (GUI) Initial Permissions for PVE

IN PRODUCTION: Because the first node in a cluster will overwrite the Permissions settings of all subsequent nodes that join the cluster, you would perform the following only on the first node.

Step 1: Datacenter > Permissions > Add > Group Permissions

e.g.:



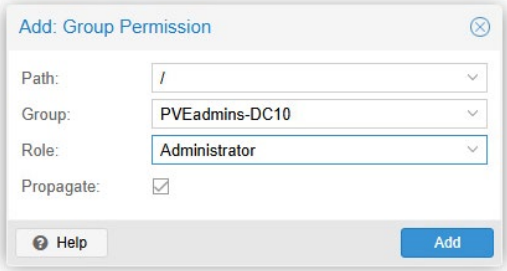
Step 2: Path: /

Step 3: Group: PVEadmins-DC10

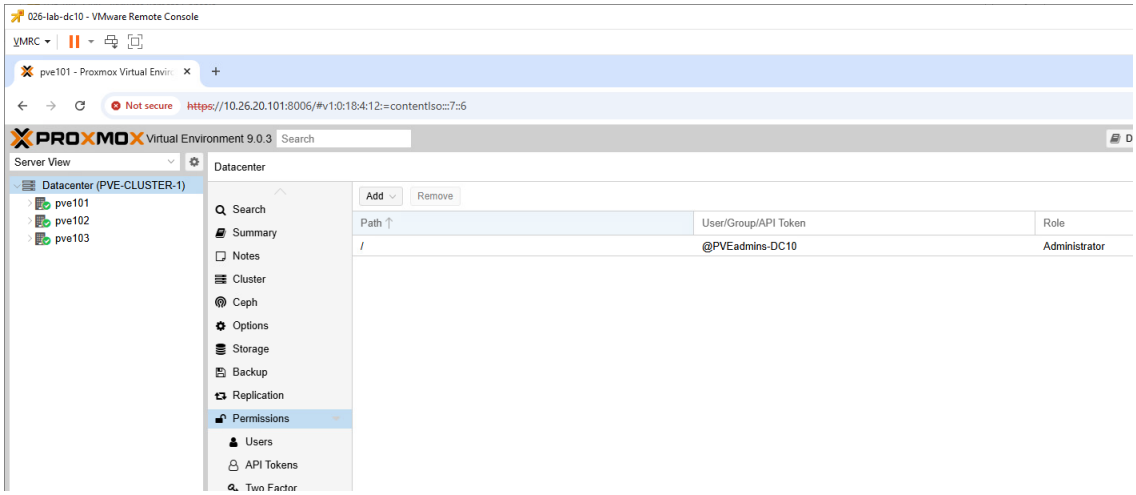
Step 4: Role: Administrator

Step 5: Add

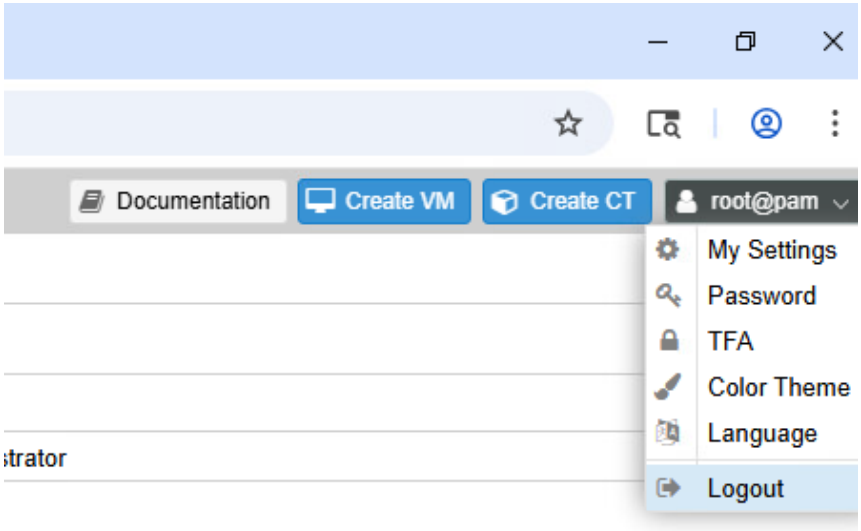
e.g.:



e.g.:

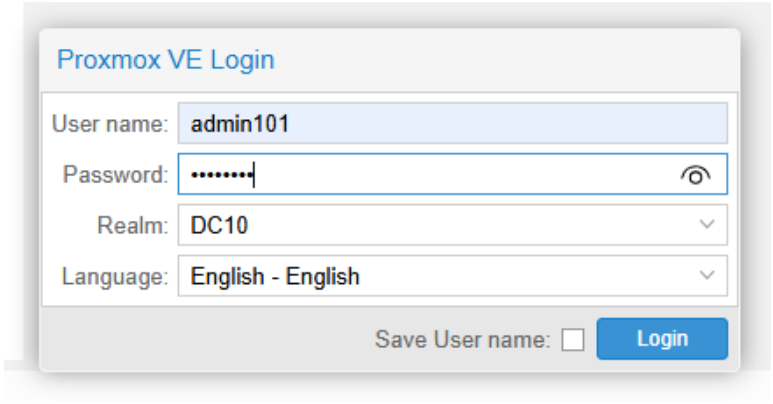


Step 6: Logout



e.g.:

Step 7: Log on



e.g.:

SBS LAB – (CLI/GUI) Notifications for PVE

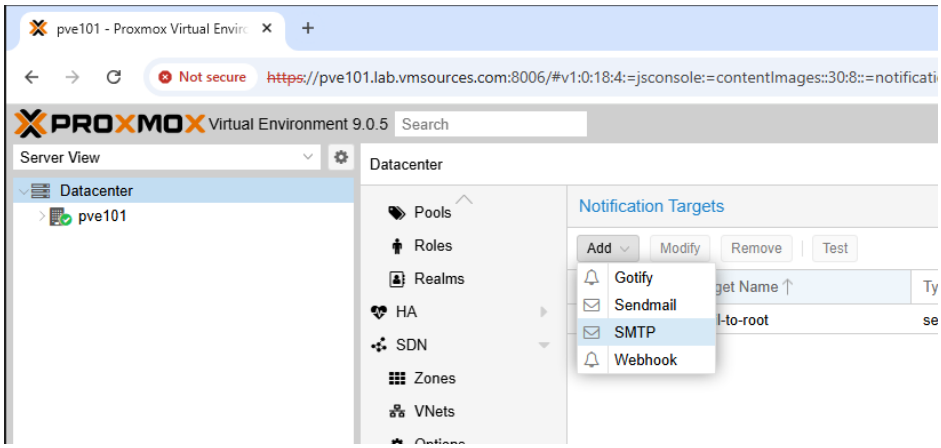
Notifications for PVE are essential. There's a ton of information online about how to use Gmail with an App password, and quite a bit of unnecessary rigamarole regarding necessary reconfiguration of Postfix.

The reality is that PVE is NOT capable of OAuth 2 integrations at the moment, so you will likely be unable to send email directly to your enterprise mail server unless you can configure it as a SMTP relay to a specific authorized IP address.

If you can't set up a SMTP relay on your mail server, you will need a mail-relay server.

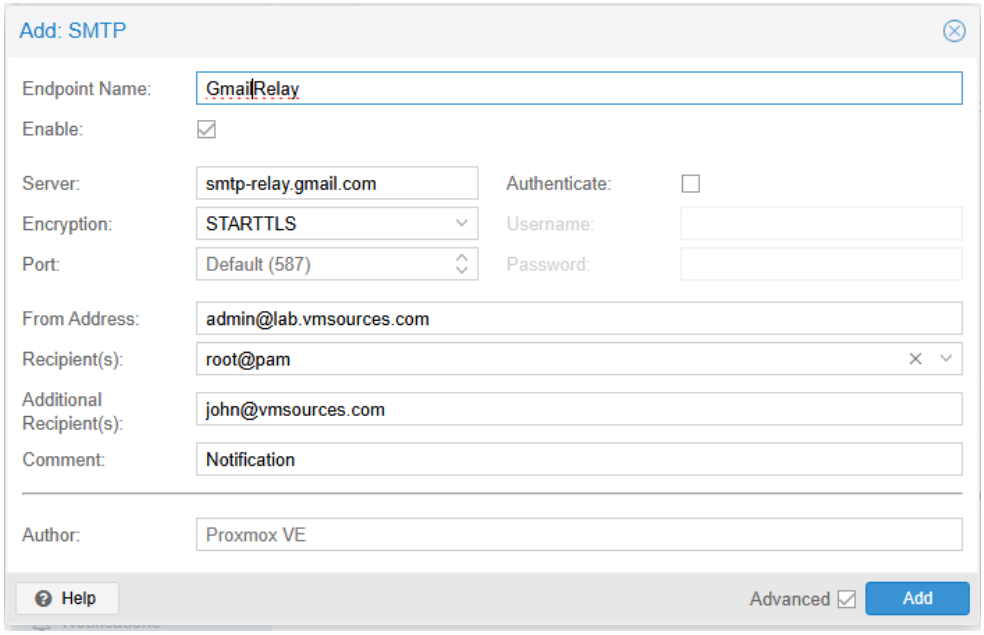
IN PRODUCTION: Because the first node in a cluster will overwrite the notification settings of all subsequent nodes that join the cluster, you would perform the following only on the first node.

1. Add a SMTP Notification Target
   Step 1: pveXYZ > Notifications > Notification Targets > Add > SMTP
   e.g.:



2. Configure SMTP Target
   Step 1: Endpoint Name: Any name (no spaces)
   Step 2: Server: Your email relay server
   Step 3: Encryption: Required encryption for your email relay server
   Step 4: Port: Required port for your email relay server
   Step 5: From Address: The address mails will be sent from
   Step 6: Recipients: Select from PVE users (set email in users)
   Step 7: Additional Recipients: Any other email addresses
   Step 8: Add

e.g.:

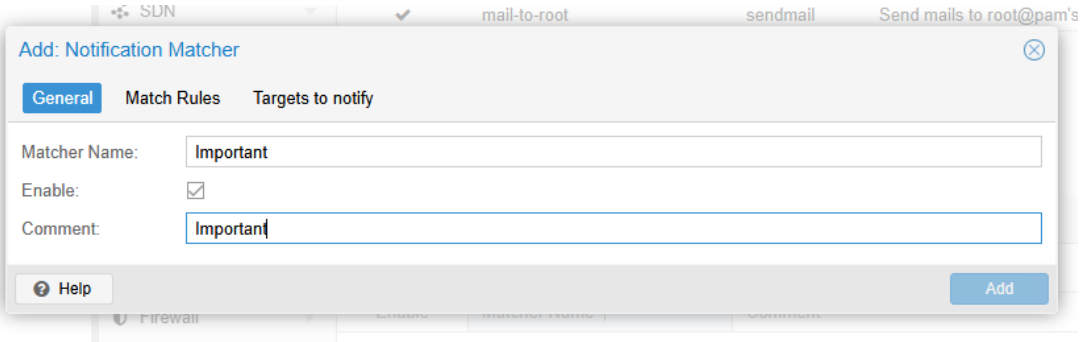


3. Configure Notification Matcher

   Step 1: pveXYZ > Notifications > Notification Matchers > Add

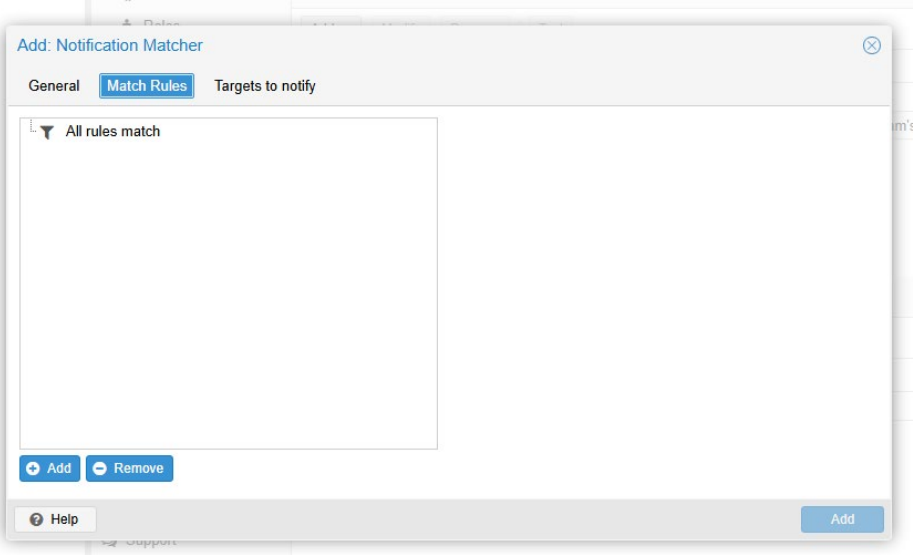
   Step 2: Matcher Name: Any name

e.g.:



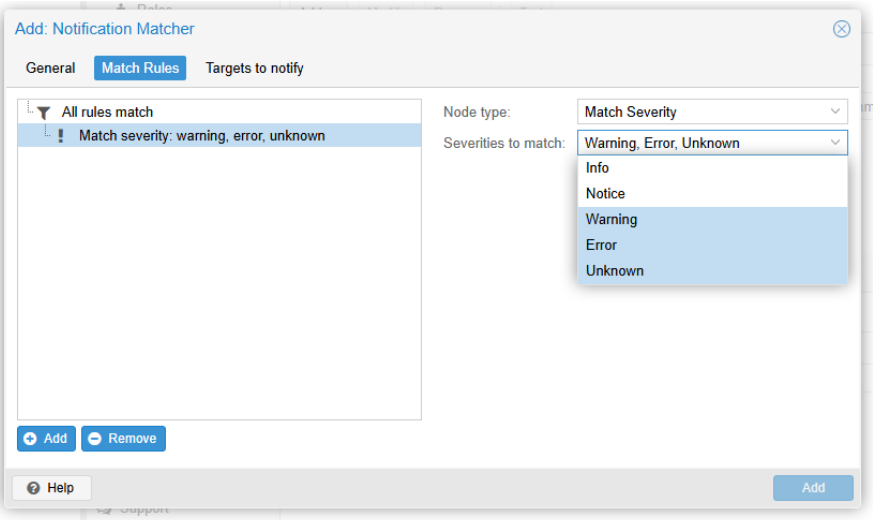
Step 3: Match Rules > Add

e.g.:



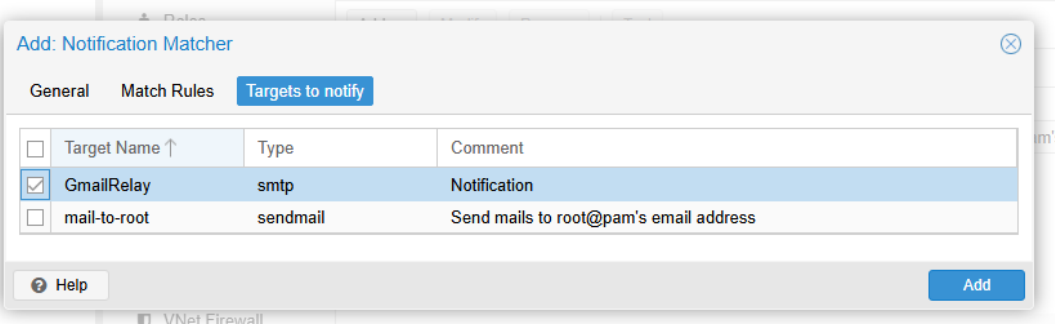
Step 4: Set: Match Severity > Warning, Error, Unknown

e.g.:



Step 5: Targets to notify > your relay server > Add

e.g.:



Networking for PVE

Out of the box, PVE networking provides only for IP connectivity and management,

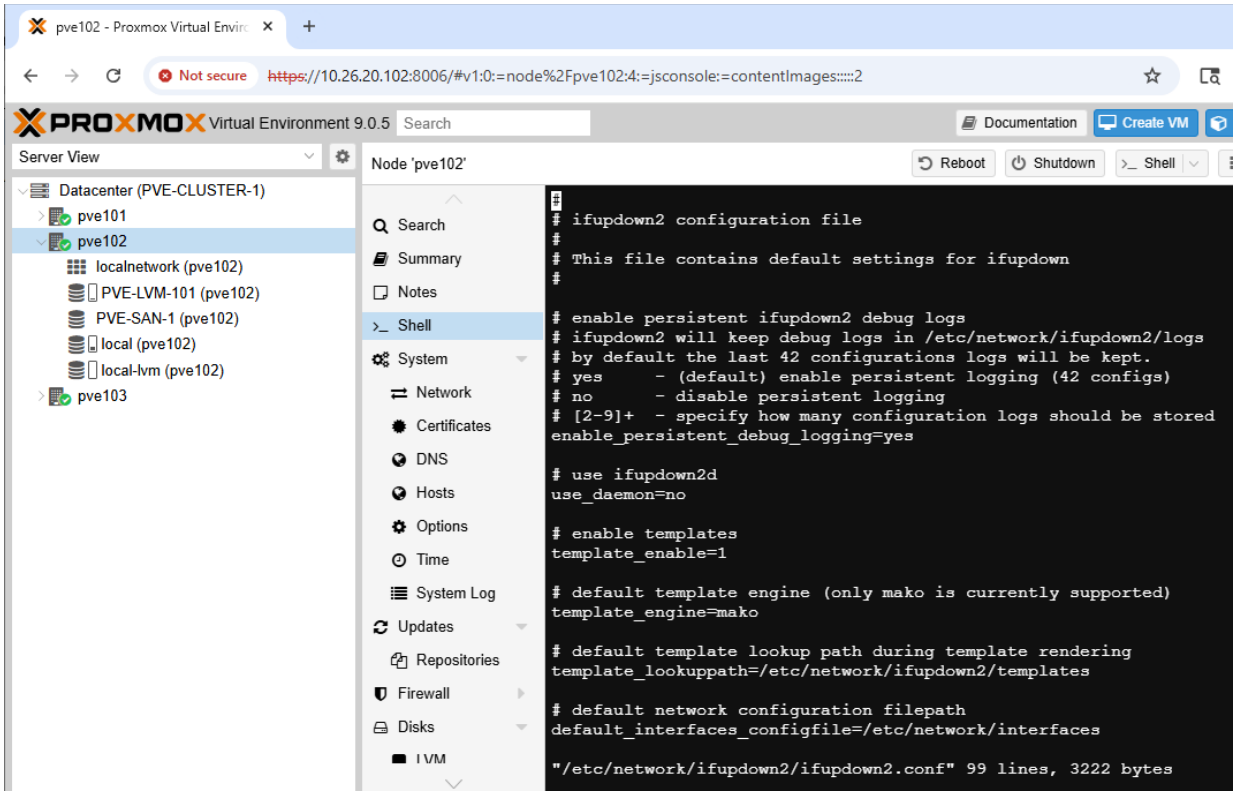
SBS LAB – (CLI) Tuning network for resiliency at reboot

In some situations, PVE has shown a tendency to 'loose' its gateway (routing) after a reboot. This is an issue with ifupdown2[v] which only affects PVE environments which are accessed via routing.

Step 1: Open a terminal to your PVE node

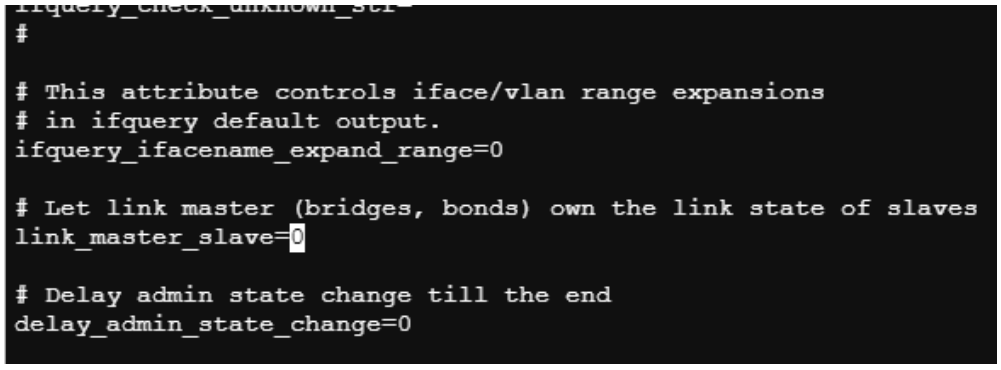
Step 2: Edit the /etc/network/ifupdown2/ifupdown2.conf file
Command #1 run: vi /etc/network/ifupdown2/ifupdown2.conf

e.g.:



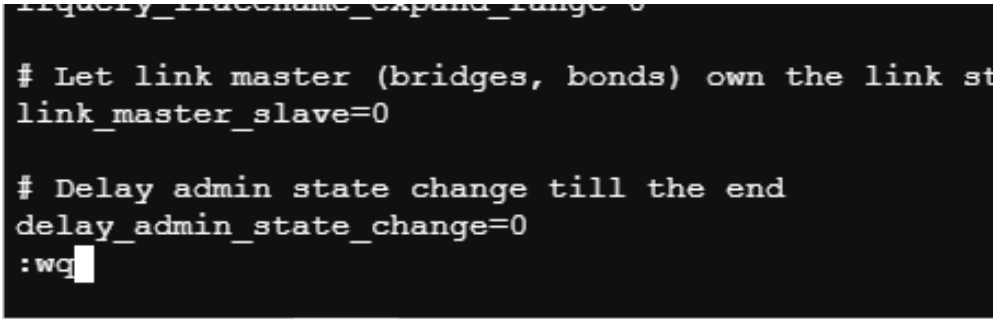
Step 3: Locate the variable 'link_master_slave=1' and change it to 'link_master_slave=0'

e.g.:



Step 4: Write and quit

e.g.:



SBS LAB – (GUI) Linux Bond for PVE

The Linux Bond is a simple and robust way of managing interfaces for load balancing and failover. In our lab environment, 4 of 6 ports which our interfaces connect to are TRUNK ports with access to VLAN 20-26. That means that only VLAN TAGGED traffic will pass through.

Our goals will be:

- Redundancy
- Diversity
- Load Balancing

We will achieve redundancy by using two (or more) interfaces on our PVE Node. We will achieve diversity by making sure that each PVE interface is connected to a different Leaf Switch (first upstream switch from PVE) and ww will achieve load balancing by selecting the appropriate policy for our network.

In this lab we are using two interfaces for the Linux Bond, two interfaces are dedicated to Jumbo Frames, and we are leaving two interfaces for use with Open vSwitch (oVS).

There are 7 modes for bonding[vi]:

Round-robin (balance-rr): Transmit network packets in sequential order from the first available network interface (NIC) slave through the last. This mode provides load balancing and fault tolerance.

Active-backup (active-backup): Only one NIC slave in the bond is active. A different slave becomes active if, and only if, the active slave fails. The single logical bonded interface's MAC address is externally visible on only one NIC (port) to avoid distortion in the network switch. This mode provides fault tolerance.

XOR (balance-xor): Transmit network packets based on [(source MAC address XOR'd with destination MAC address) modulo NIC slave count]. This selects the same NIC slave for each destination MAC address. This mode provides load balancing and fault tolerance.

Broadcast (broadcast): Transmit network packets on all slave network interfaces. This mode provides fault tolerance.

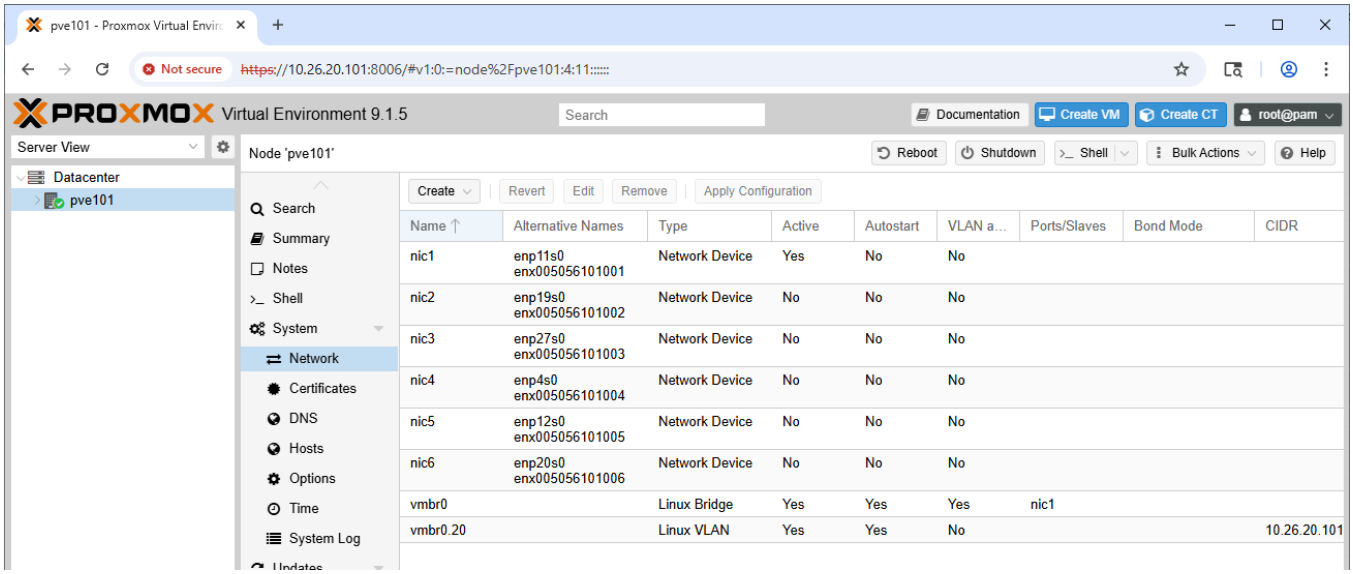
IEEE 802.3ad Dynamic link aggregation (802.3ad)(LACP): Creates aggregation groups that share the same speed and duplex settings. Utilizes all slave network interfaces in the active aggregator group according to the 802.3ad specification.

Adaptive transmit load balancing (balance-tlb): Linux bonding driver mode that does not require any special network-switch support. The outgoing network packet traffic is distributed according to the current load (computed relative to the speed) on each network interface slave. Incoming traffic is received by one currently designated slave network interface. If this receiving slave fails, another slave takes over the MAC address of the failed receiving slave.

Adaptive load balancing (balance-alb): Includes balance-tlb plus receive load balancing (rlb) for IPV4 traffic, and does not require any special network switch support. The receive load balancing is achieved by ARP negotiation. The bonding driver intercepts the ARP Replies sent by the local system on their way out and overwrites the source hardware address with the unique hardware address of one of the NIC slaves in the single logical bonded interface such that different network-peers use different MAC addresses for their network packet traffic.

> IN PRODUCTION:
>
> Active-backup is safest when bandwidth is not a concern. **We use this in our lab environment for maximum compatibility with upstream virtual switches and nested PVE environments.**
>
> LACP is appropriate when the corresponding upstream switches are also configured for LACP. We use this in PROD environments where PVE will be connected directly to the network and upstream switch configuration can be verified.
>
> Round-Robbin works well however packets may be delivered non-synchronously (effectively Delayed-ACK) and the upstream switches need to be able to accommodate that.

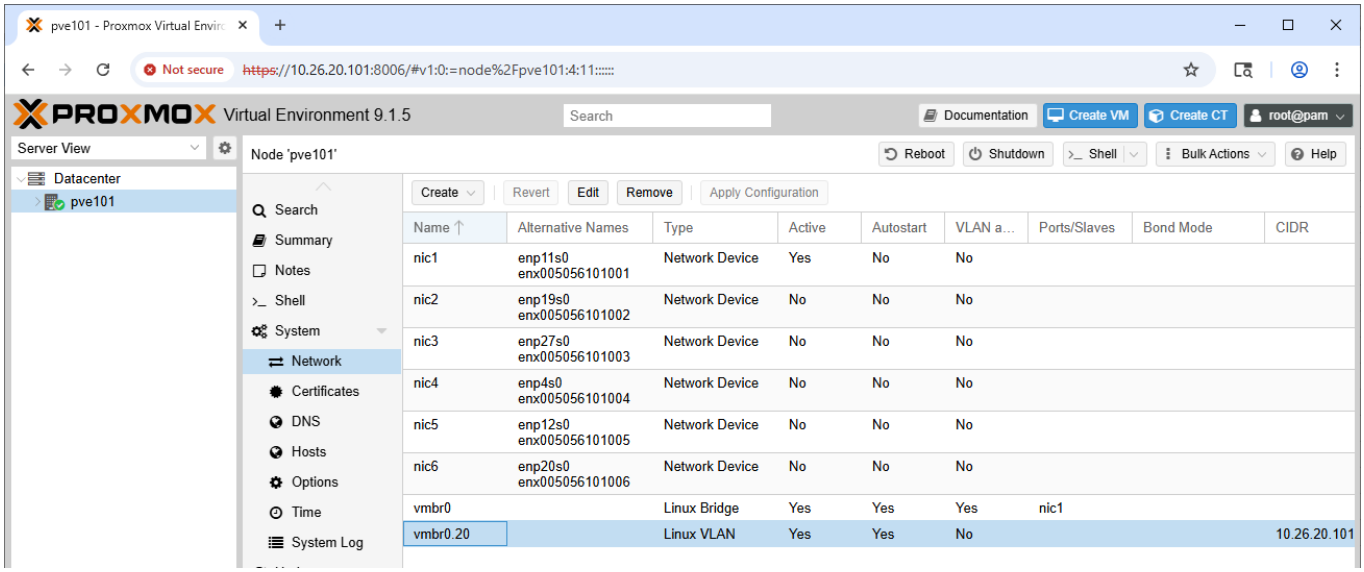
1. The first thing we are going to do is clear out the basic network for management and trunking provided by the installer.
   Step 1: Go to: pveXYX > Network
   e.g.:



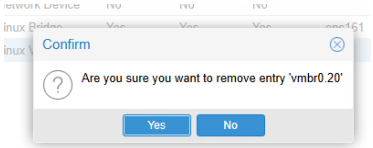
NOTE : One of the notable improvements in PVE 9 is that the main Networking screen shows the Alternative Names, which also include the MAC address, so you can more correctly associate the interfaces as you consume them

Step 2: Remove the Linux VLAN

e.g.:

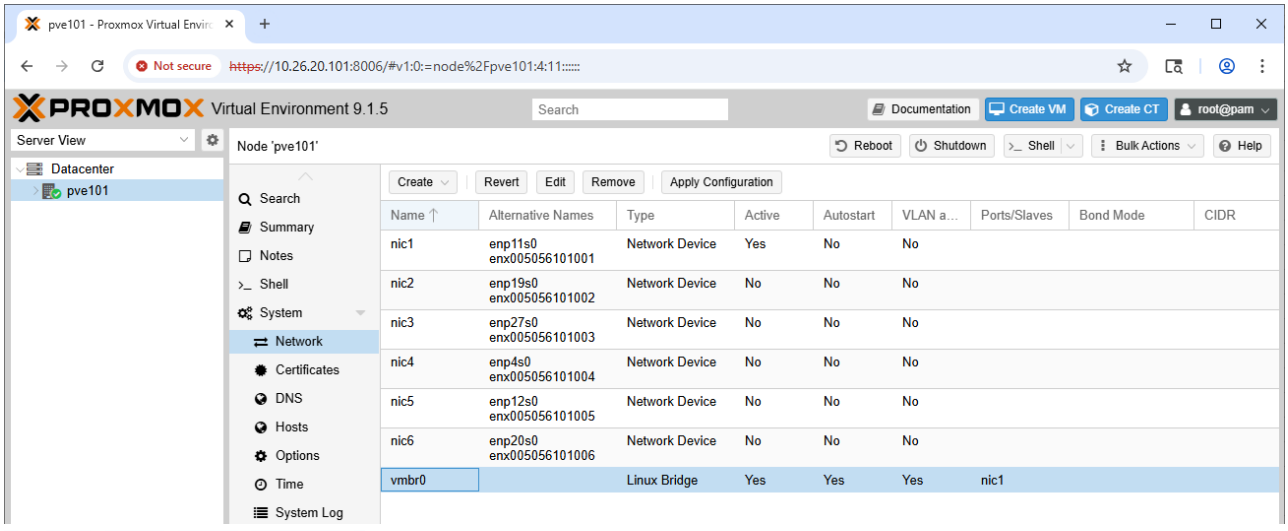


e.g.:

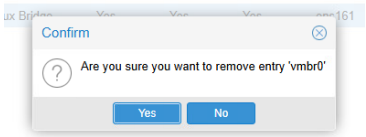


Step 3: Remove the Linux Bridge

e.g.:



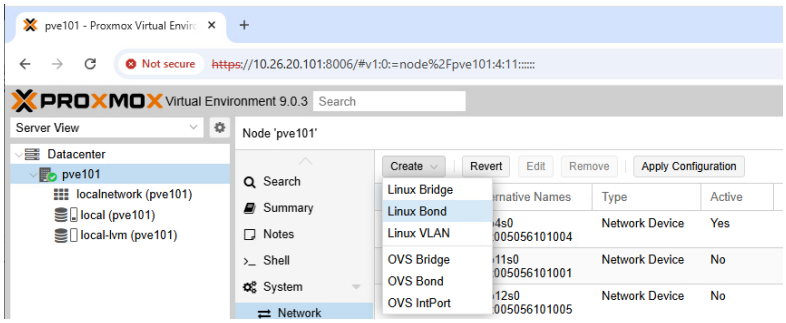
e.g.:



2. We are going to create a Linux Bond for the interfaces we will be using as management and for VM trunking

   Step 1: Create > Linux Bond

   e.g.:



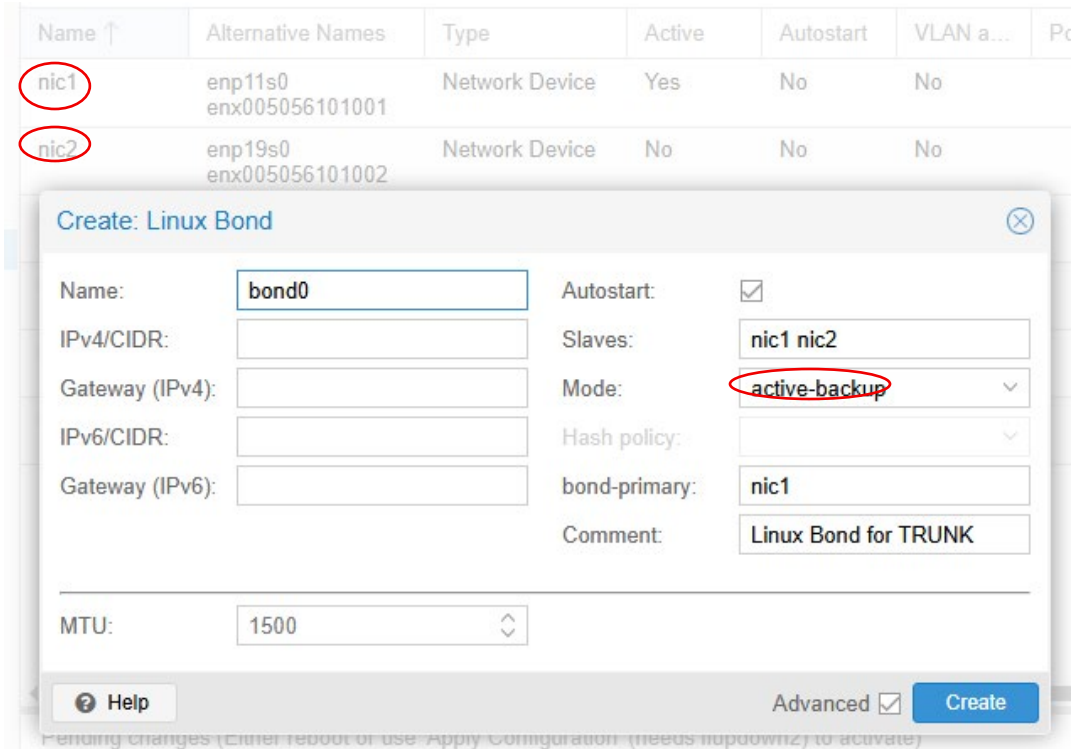
   NOTE : We are going to use nic1 and nic2

   Step 2: The bond will be named automatically, don't change it. Specify as Slaves two interfaces which can access the management network

   Step 3: Mode: **Active-Backup**

   Step 4: Add comment for purpose: Linux Bond for TRUNK

   e.g.:



SBS LAB –(GUI) Linux Bridge for TRUNK

The Linux Bridge is a form of virtual switch, allowing access to networking. The Linux bridge can be created on a single VLAN, or a range of VLANs can be specified. This is also a prerequisite for creating our Management Interface on VLAN 20

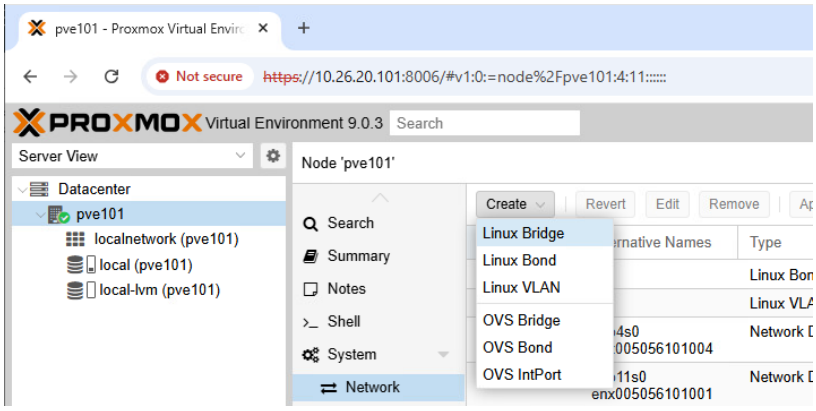
Variables for this lab (or other as appropriate for your environment):

- VLAN IDs: 20-26

1. Now create a Linux Bridge for VM trunking

   Step 1: Create > Linux Bridge

   e.g.:



   Step 2: Name: vmbr0

   Step 3: Autostart: checked

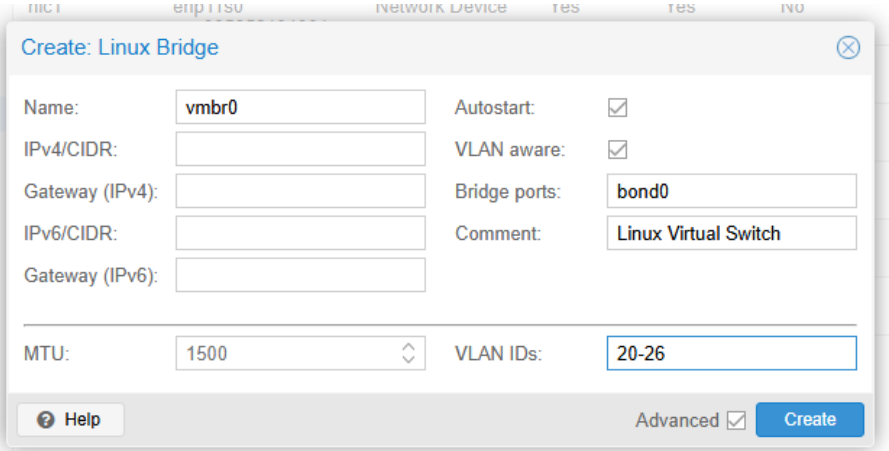
   Step 4: VLAN aware: checked

   Step 5: Bridge ports: bond0

   Step 6: Add comment for purpose: Linux Virtual Switch

   Step 7: VLAN IDs: 20-26 (The VLANs that are trunked to PVE)

   e.g.:



SBS LAB – (GUI) Linux VLAN for Management

A Linux VLAN is a simple component used to assign an IP address (and possibly Gateway) to an upstream Bond, Bridge, or Interface. Here, we are going to use it for our management IP for PVE.

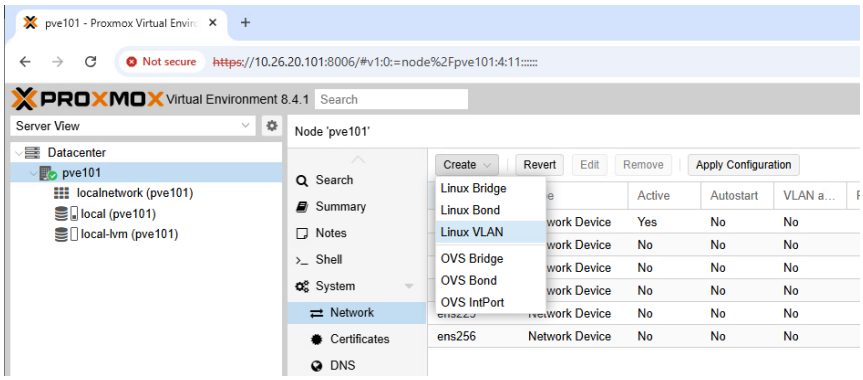
Variables for this lab(or other as appropriate for your environment):

- VLAN name: vmbr0.20
- MGMT IP: 10.26.20.XYZ/24
- GW: 10.26.20.1

1. Now we need a management IP, which we will achieve using a Linux VLAN

Step 5: Create a Linux VLAN for Management using bond0

e.g.:



NOTE : Naming is very important here. We need to name the Linux VLAN with the name of the bridge followed by a [.] and then the VLAN ID. E.g.: vmbr0.20 You will then see that the VLAN Tag is automatically set.

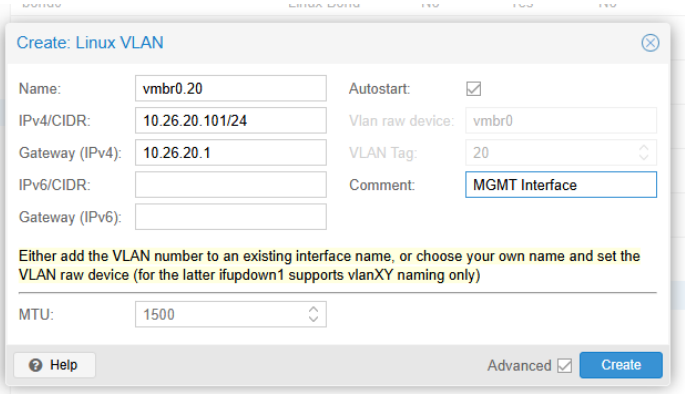
Step 6: Name the bond: vmbr0.20

Step 7: Enter the IP: 10.26.20.XYZ/24

Step 8: Enter the Gateway: 10.26.20.1

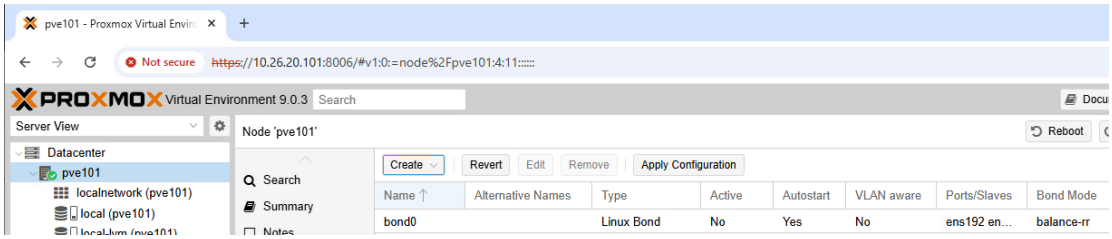
Step 9: Add comment for purpose: MGMT IP

e.g.:

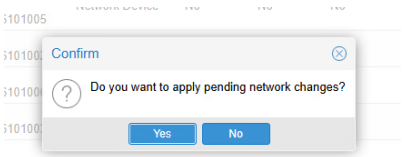


Step 5: Now Apply Configuration to test your work

e.g.:



e.g.:



Storage Networking for PVE

SBS LAB –(GUI) iSCSI Interface Config for PVE with Jumbo Frames

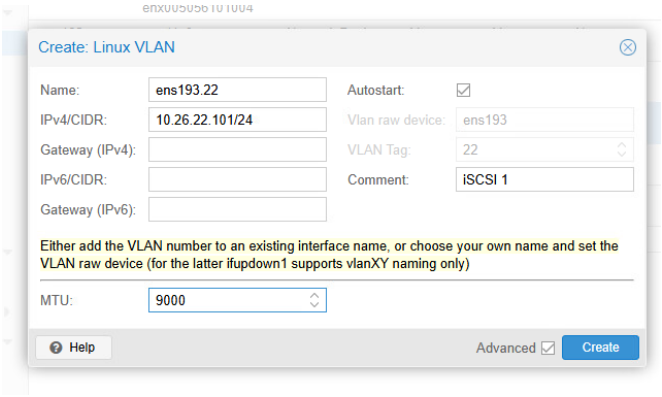
In this lab we are going to set up the interfaces for iSCSI. Our iSCSI ports on the switch are configured as ACCESS ports meaning that the client does not need to specify a VLAN ID, and only the VLAN which is configured for that port is accessible.

IN PRODUCTION:

There's nothing wrong with running iSCSI on a VLAN, just 4 bytes more overhead per packet. The only difference in configuration would be setting the MTU property without IP on the interface and you would create a Linux VLAN for the iSCSI VLAN and IP.

Since some SANs do not support VLAN configuration, and because there is a tiny bit more overhead, we prefer to run iSCSI on ACCESS ports.

Example iSCSI config with VLAN (***do not use in class***):



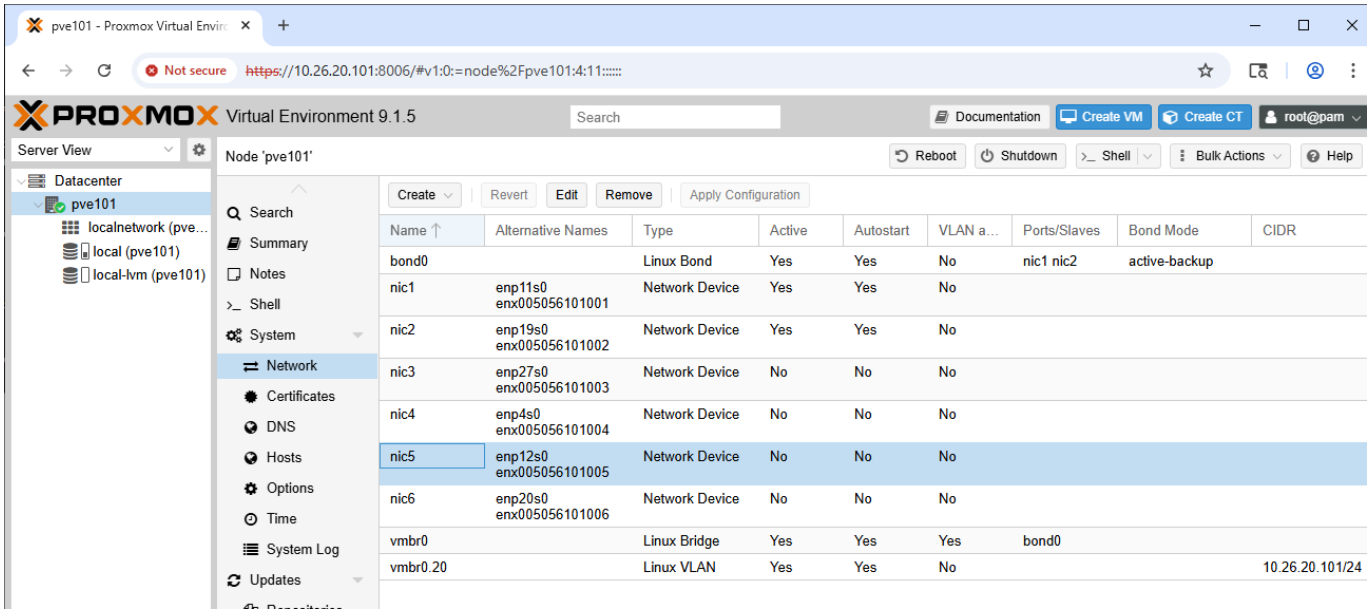
Variables for this lab (or other as appropriate for your environment):

- nic5 IP: 10.26.22.XYZ/24
- nic5 comment: iSCSI 1
- MTU: 9000
- nic6 IP: 10.26.23.XYZ/24
- nic6 comment: iSCSI 2
- MTU: 9000

1. iSCSI Interfaces

Step 1: Choose the iSCSI interface which has a MAC addresses (last digits of Alternative Names) ending in 05 (corresponding to VLAN 22) and select Edit.

e.g.:



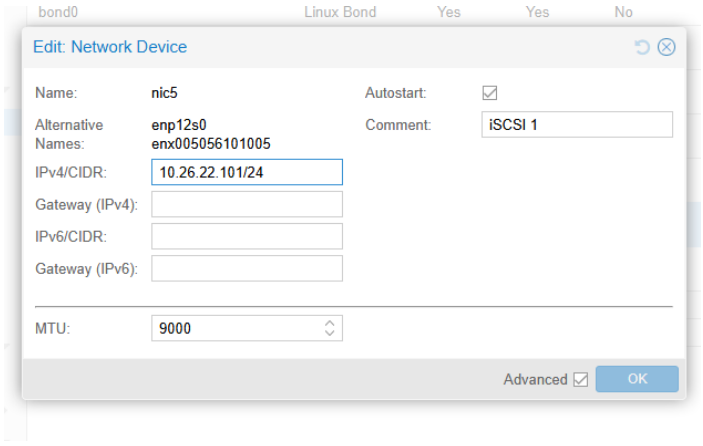
Step 2: Autostart: checked

Step 3: IP: 10.26.22.XYZ/24

Step 4: Set MTU to 9000

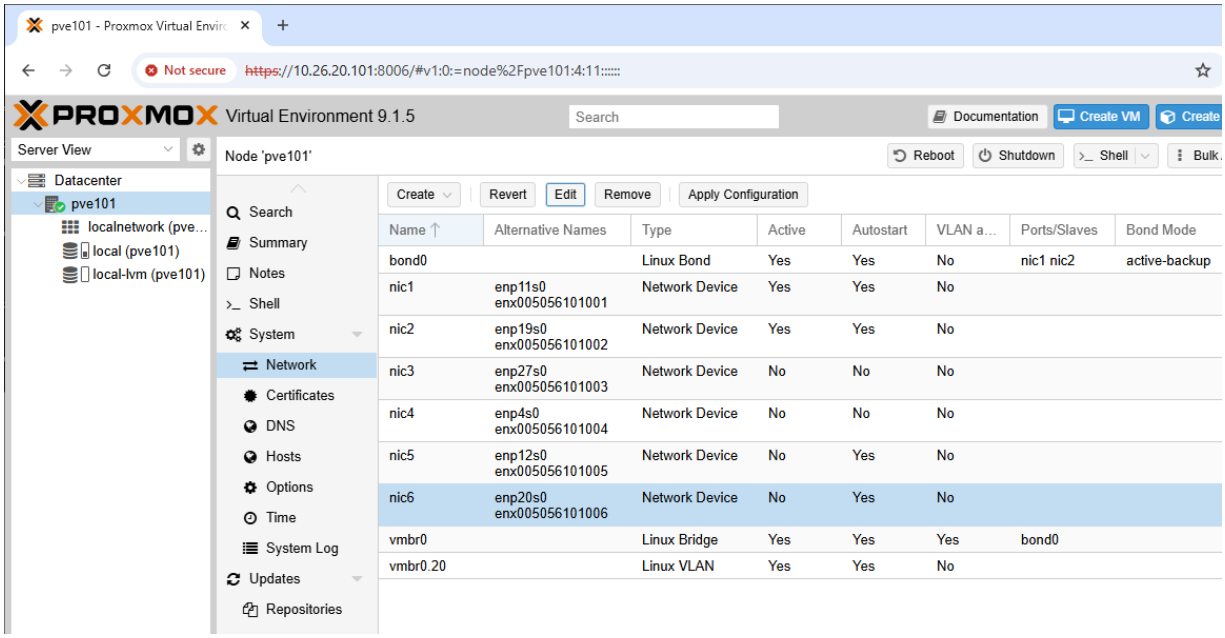
Step 5: Add comment for purpose: iSCSI 1

e.g.:



Step 6: Now choose the iSCSI interface which has a MAC addresses (last digits of Alternative Names) ending in 06 (corresponding to VLAN 23) and select Edit.

e.g.:



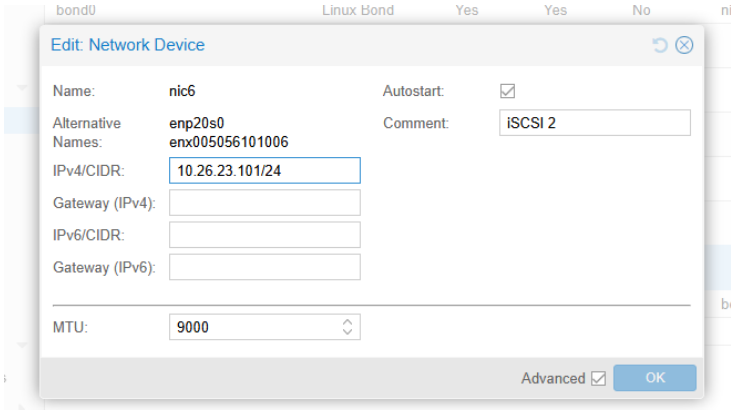
Step 7: Autostart: checked

Step 8: IP: 10.26.23.XYZ/24

Step 9: Set MTU to 9000

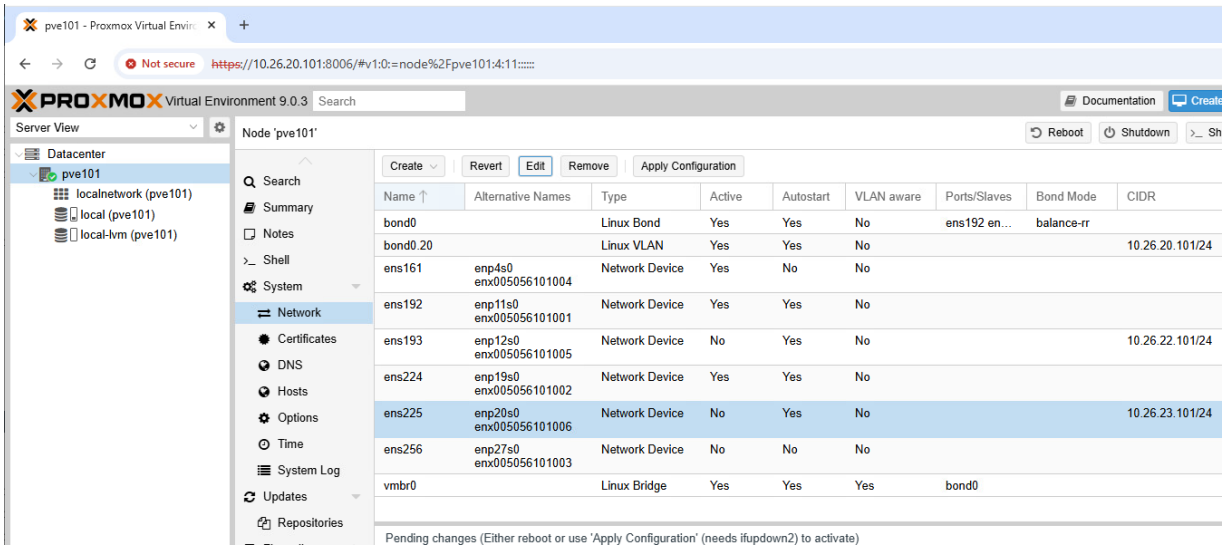
Step 10: Add comment for purpose: iSCSI 2

e.g.:

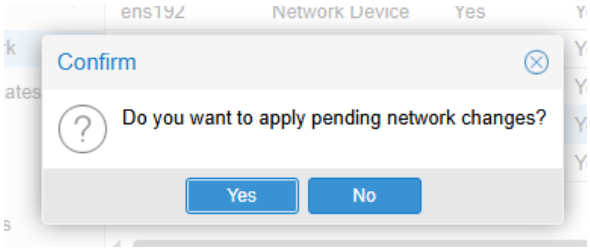


Step 11: Apply changes.

e.g.:



e.g.:

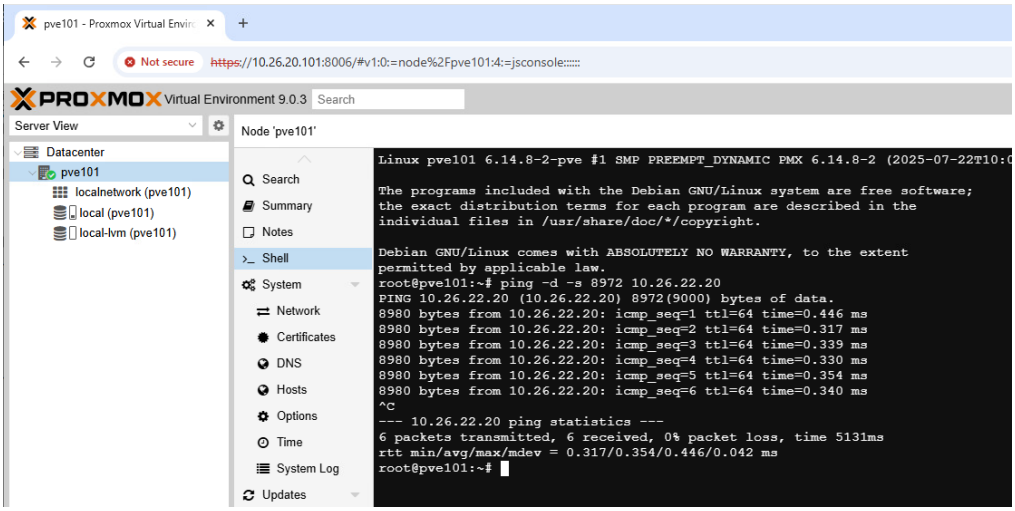


4. Now let's test connectivity to our SAN

Step 11: Go to the Shell

Command #1 run: ping -d -s 8972 10.26.22.20

5.

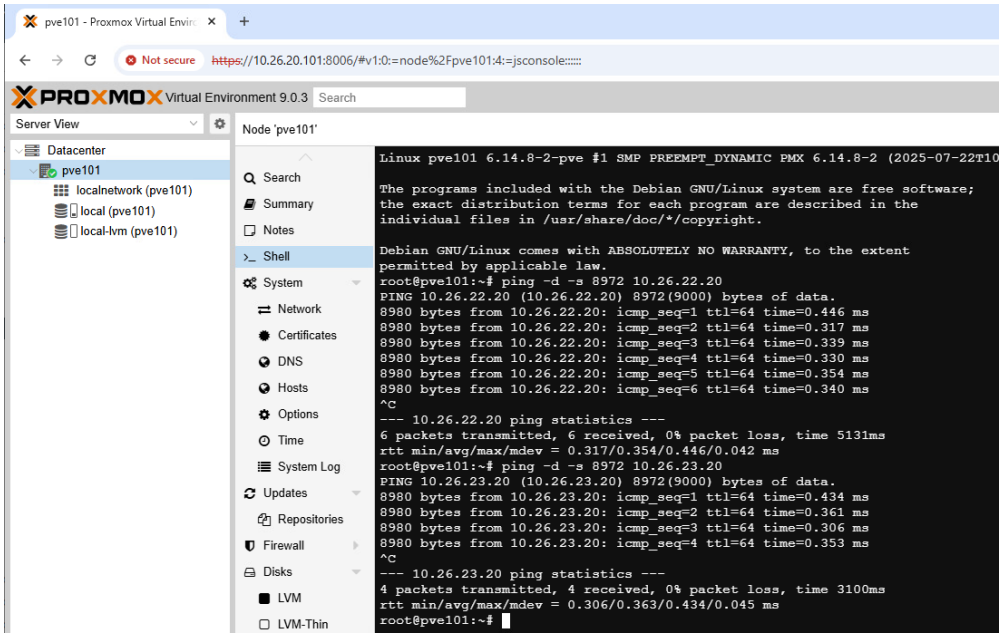


NOTE : You will have to stop the ping with a [CTRL]+[c]

Step 11: Now do the same for the other interface

Command #1 run: ping -d -s 8972 10.26.23.20

Step 12:



NOTE : You will have to stop the ping with a [CTRL]+[c]

CHECKPOINT: We have just configured two interfaces for iSCSI and Jumbo Frames. In executing the ping command, the [-d] option does not allow the jumbo frame to be segmented and the [-s] option specifies the size of the payload which the maximum possible payload for a 9000 byte network is 8972 to which headers are added when the frame is transmitted.

DIAGNOSTICS:

In the event that either of the ping commands do not respond, try the ping without the options:

ping 10.26.22.20

ping 10.26.23.20

If those respond, Jumbo Frames are not configured correctly on your network or SAN

In the event that no pings are received, either Jumbo or normal, check the interfaces configured match interfaces which are connected to iSCSI ACCESS ports.

iSCSI SAN Storage for PVE

iSCSI Multipath is a necessity for production environments because it facilitates more traffic to and from your SAN. Unfortunately, the traditional network load balancing methods available in a Linux Bond are not suitable for iSCSI traffic, so we must configure iSCSI independently using the CLI and GUI for correct multipath operation.

iSCSI Configuration can vary greatly based on the environment. As a Best Practice, Proxmox recommends that all separate iSCSI paths, also be on separate subnets:

PVE iSCSI NIC #1 (10.26.22.XYZ/24) > SAN NIC #1 (10.26.22.20/24)

PVE iSCSI NIC #2 (10.26.23.XYZ/24) > SAN NIC #2 (10.26.23.20/24)

In the event that this is not possible, then some form of 'binding' is required to properly distinguish NIC/Target relationships over the network. This is covered at the end of the section.

The way you connect to your iSCSI SAN depends on several factors including:

- How the iSCSI SAN presents its targets
- How many interfaces the SAN has
- How the network is configured
- How many iSCSI interfaces the PVE node has

SBS LAB – (CLI) Identify the way your SAN presents targets

Key to understanding what will be required to connect to your iSCSI SAN correctly is the command:

iscsiadm -m discovery -t sendtargets -p <IP of SAN>

A SAN which presents the same target on multiple distinct networks is considered Best Practices and will generally be easier to configure in PVE.

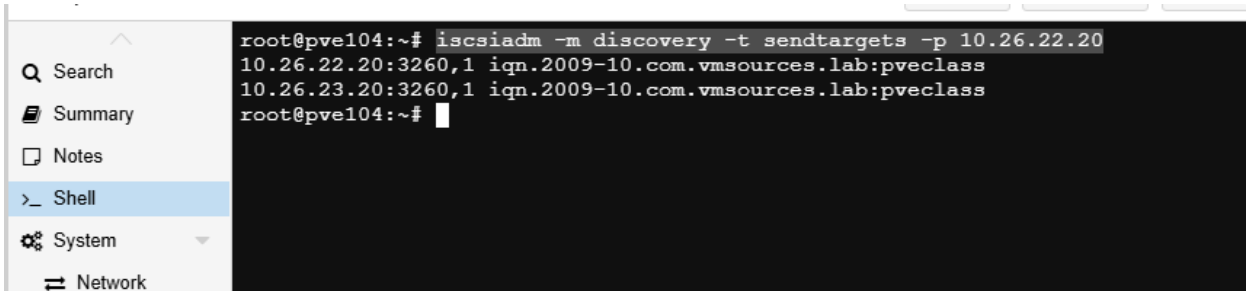
A SAN which presents the same target multiple times on the same network will require further configuration to specify which iSCSI interface on PVE (hopefully you have more than one) will communicate with which IP/target on the SAN.

1. Let's test a SAN with interfaces configured on separate unique networks

   Step 1: Go to: pveXYX > Shell

   Command #1 run: iscsiadm -m discovery -t sendtargets -p 10.26.22.20

   e.g.:

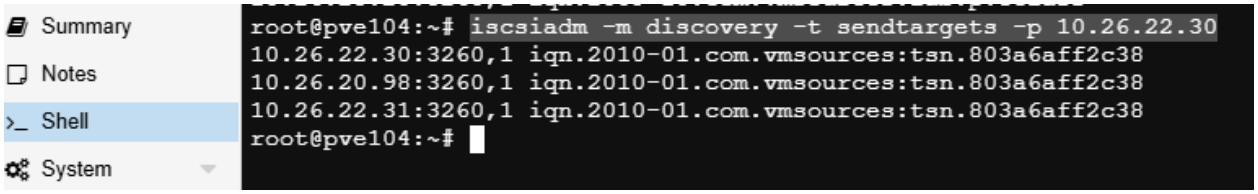


   NOTE : This SAN presents the same target on two distinct networks: 10.26.22.20 and 10.26.23.20, per Best Practices

   Step 2: Now let's test a SAN with interfaces configured on the same network

   Command #1 run: iscsiadm -m discovery -t sendtargets -p 10.26.22.30

   e.g.:



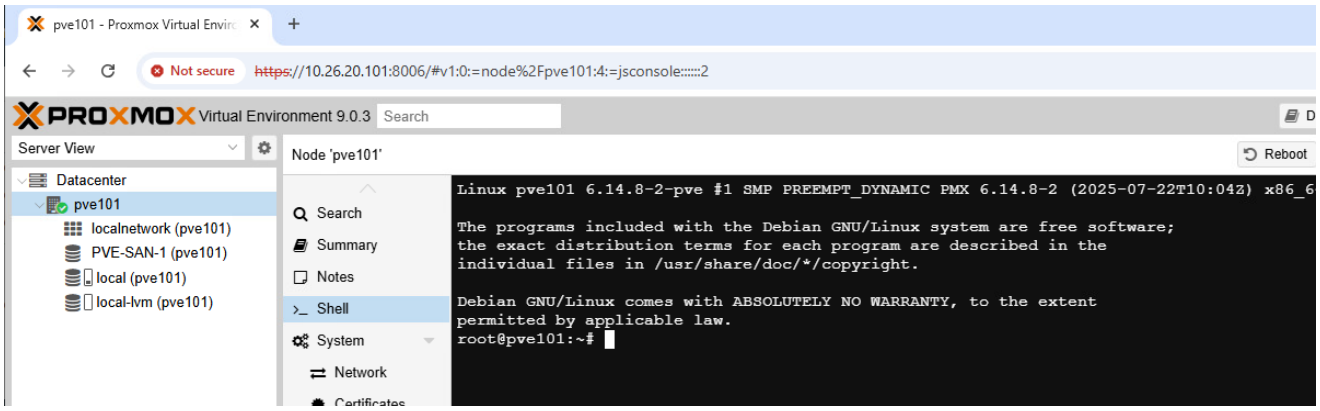
   NOTE : This SAN presents the same target twice on the same network: 10.26.22.30 and 10.26.22.31 (ignore the 10.26.20.98 entry as that is the management interface) and is typical of many "legacy" or inexpensive iSCSI devices.

SBS LAB – (CLI/GUI) Installing and Configuring Prerequisites for iSCSI Multipath

1. First, we need to change some settings and install prerequisites

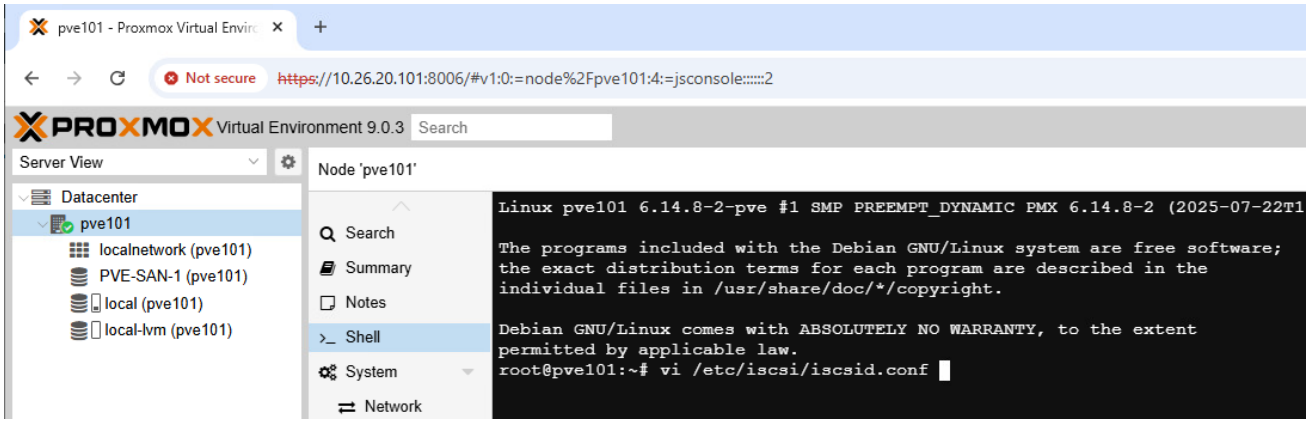
Step 3: Go to: pveXYX > Shell

e.g.:



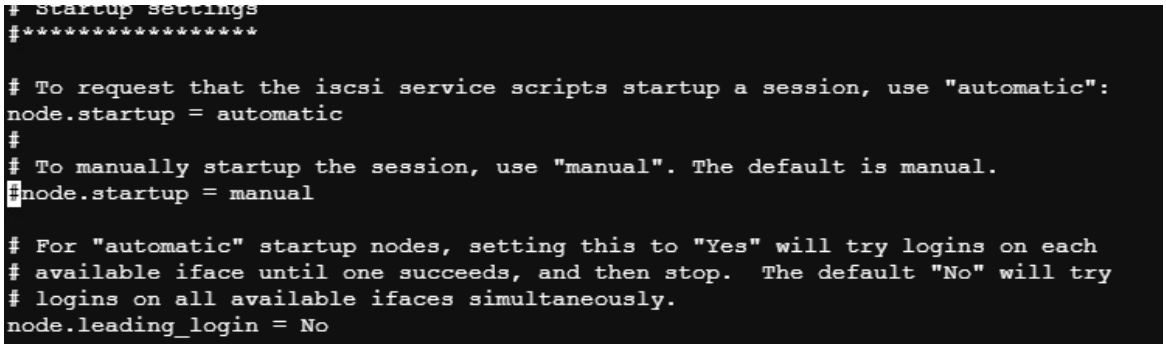
Command #1 run: vi /etc/iscsi/iscsid.conf

e.g.:



Step 2: Comment [#]: node.startup = manual and uncomment: node.startup = automatic

e.g.:



Step 3: Change: node.session.timeo.replacement_timeout = 120 to
node.session.timeo.replacement_timeout = 15

e.g.:

```
# See the iSCSI README's Advanced Configuration section for tips
# on setting timeouts when using multipath or doing root over iSCSI.
#
# To specify the length of time to wait for session re-establishment
# before failing SCSI commands back to the application when running
# the Linux SCSI Layer error handler, edit the line.
# The value is in seconds and the default is 120 seconds.
# Special values:
# - If the value is 0, IO will be failed immediately.
# - If the value is less than 0, IO will remain queued until the session
# is logged back in, or until the user runs the logout command.
node.session.timeo.replacement_timeout = 15

# To specify the time to wait for login to complete, edit the line.
# The value is in seconds and the default is 15 seconds.
```

Step 4: Write and quit

e.g.:

```
# the Linux SCSI Layer error handler, edit the line.
# The value is in seconds and the default is 120 seconds.
# Special values:
# - If the value is 0, IO will be failed immediately.
# - If the value is less than 0, IO will remain queued until the session
# is logged back in, or until the user runs the logout command.
node.session.timeo.replacement_timeout = 15

# To specify the time to wait for login to complete, edit the line.
# The value is in seconds and the default is 15 seconds.
:wq
```

Step 5: Restart iSCSI Service

Command #1 run: systemctl restart iscsi.service

e.g.:

```
permitted by applicable law.
root@pve101:~# systemctl restart iscsi.service
root@pve101:~#
```

Step 6: Now, install Multipath Tools

Command #1 run: apt install multipath-tools

e.g.:

```
Linux pve101 6.14.8-2-pve #1 SMP PREEMPT_DYNAMIC PMX 6.14.8-2 (2025-07-22T10:04Z) 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.
root@pve101:~# apt install multipath-tools
```

Step 1: Enter [Y]

e.g.:

```
root@pve101:~# apt install multipath-tools
Installing:
  multipath-tools

Installing dependencies:
  kpartx  libsgutils2-1.48  sg3-utils  sg3-utils-udev

Suggested packages:
  multipath-tools-boot

Summary:
  Upgrading: 0, Installing: 5, Removing: 0, Not Upgrading: 0
  Download size: 1,549 kB
  Space needed: 5,140 kB / 10.1 GB available

Continue? [Y/n] Y
```

NOTE : Install recommended package: multipath-tools-boot

Step 7: Install multipath-tools-boot

Command #1 run: apt install multipath-tools-boot

e.g.:

```
root@pve101:~# apt install multipath-tools-boot
Installing:
  multipath-tools-boot

Summary:
  Upgrading: 0, Installing: 1, Removing: 0, Not Upgrading: 32
  Download size: 9,636 B
  Space needed: 34.8 kB / 7,677 MB available

Get:1 http://deb.debian.org/debian trixie/main amd64 multipath-tools-boot all 0.11.1-2 [9,636 B]
Fetched 9,636 B in 0s (329 kB/s)
Selecting previously unselected package multipath-tools-boot.
(Reading database ... 62175 files and directories currently installed.)
Preparing to unpack .../multipath-tools-boot_0.11.1-2_all.deb ...
Unpacking multipath-tools-boot (0.11.1-2) ...
Setting up multipath-tools-boot (0.11.1-2) ...
Processing triggers for initramfs-tools (0.148.3) ...
update-initramfs: Generating /boot/initrd.img-6.17.4-2-pve
Running hook script 'zz-proxmox-boot'..
Re-executing '/etc/kernel/postinst.d/zz-proxmox-boot' in new private mount namespace..
No /etc/kernel/proxmox-boot-uuids found, skipping ESP sync.
root@pve101:~#
```

Step 8: Enable multipath at startup

Command #1 run: systemctl enable multipathd

e.g.:

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY,
permitted by applicable law.
root@pve101:~# systemctl enable multipathd
root@pve101:~#
```

Step 9: Start multipath

Command #1 run: systemctl start multipathd

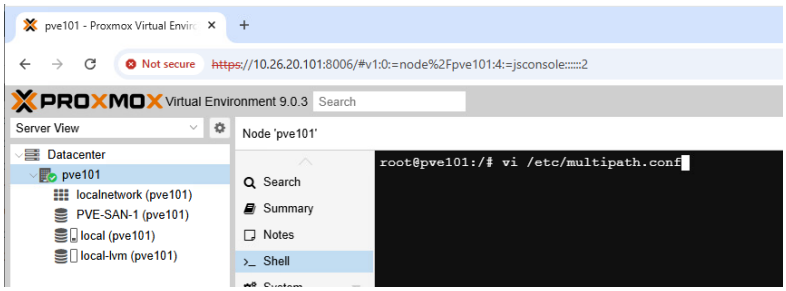
e.g.:

```
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY
permitted by applicable law.
root@pve101:~# systemctl enable multipathd
root@pve101:~# systemctl start multipathd
root@pve101:~#
```

Step 10: Finally, we can fine-tune our iSCSI Multipath by creating the file: /etc/multipath.conf

Command #1 run: vi /etc/multipath.conf

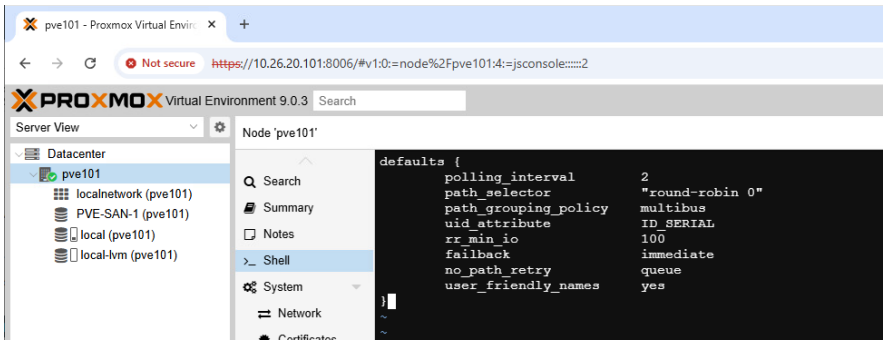
e.g.:



Step 11: Paste the following into the file[vii]

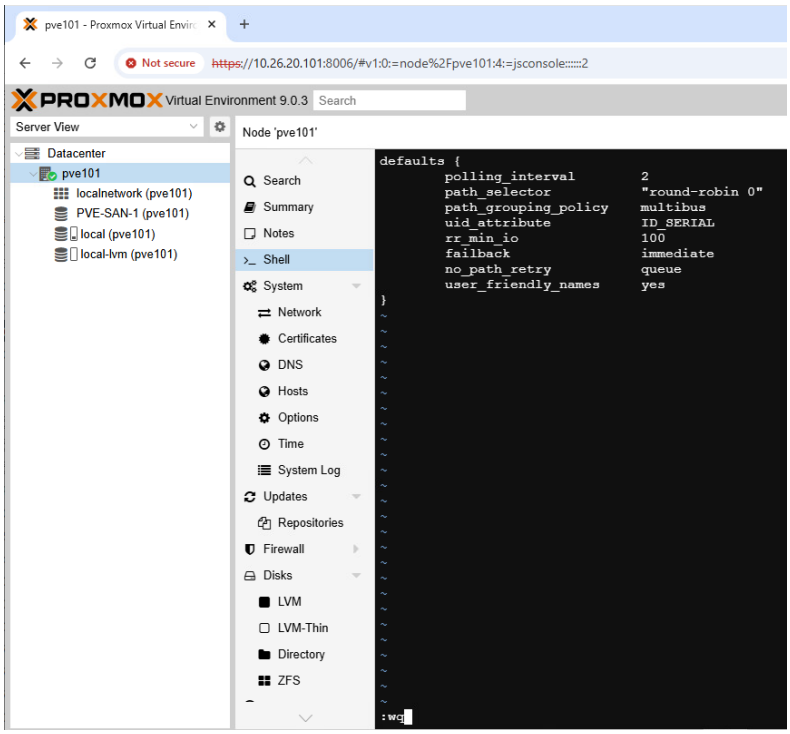
```
defaults {
        polling_interval        2
        path_selector           "round-robin 0"
        path_grouping_policy    multibus
        uid_attribute           ID_SERIAL
        rr_min_io               100
        failback                immediate
        no_path_retry           queue
        user_friendly_names     yes
}
```

e.g.:



Step 12: Write and quit

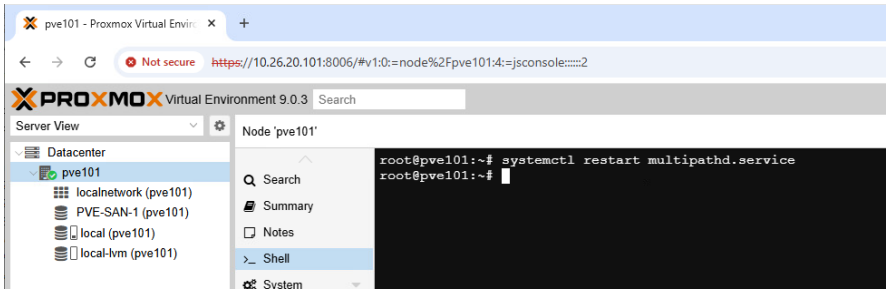
e.g.:



Step 13: Activate these settings

Command #1 run: systemctl restart multipathd.service

e.g.:



IN PRODUCTION:

You can add significant vendor-specific tuning as well as blacklist and whitelist entries to the /etc/multipath.conf file:

Backlist/whitelist entries[viii]*:*

```
blacklist {
        wwid .*
}

blacklist_exceptions {
        wwid "3600144f028f88a0000005037a95d0001"
        wwid "3600144f028f88a0000005037a95d0002"
}
```

Alias entries[ix]*:*

```
multipaths {
multipath {
        wwid "3600144f028f88a0000005037a95d0001"
        alias mpath0
}
multipath {
        wwid "3600144f028f88a0000005037a95d0002"
        alias mpath1
}
}
```

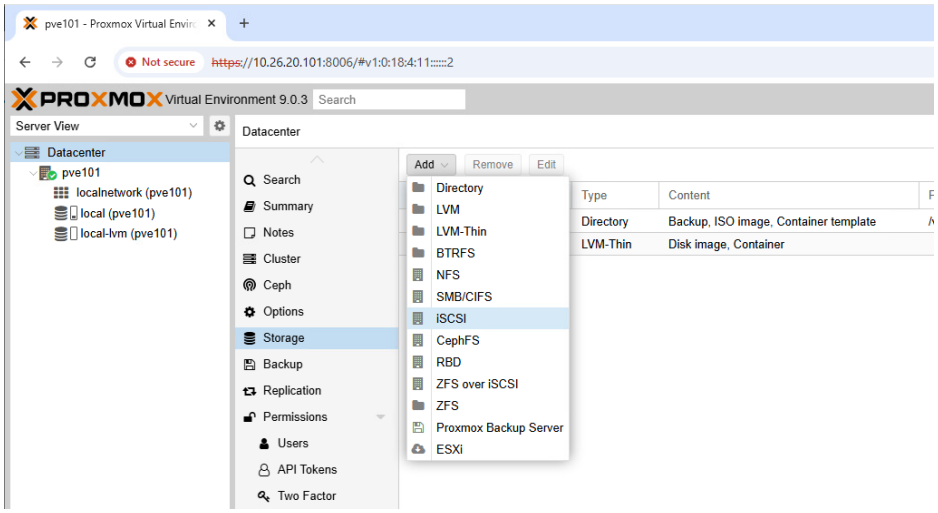
SBS LAB – (CLI/GUI) iSCSI Multipath with Network Separation

This SBS LAB and the next are either/or because the server network configuration differs from the network perspective. This is the lab we have set up for in previous SBS LABs, so this is the lab we will be following.

1. Now we are going to the GUI and connect to our SAN. The SAN will report the available portals for connection to PVE

   Step 1: Go to: Datacenter > Storage > Add > iSCSI

   e.g.:



IN PRODUCTION:

While this particular aspect of configuring iSCSI storage will be copied across the cluster, it is important to perform it individually on each PVE node to **achieve iSCSI discovery, which will then allow configuration of iSCSI multipath which must be done on each PVE node individually.**

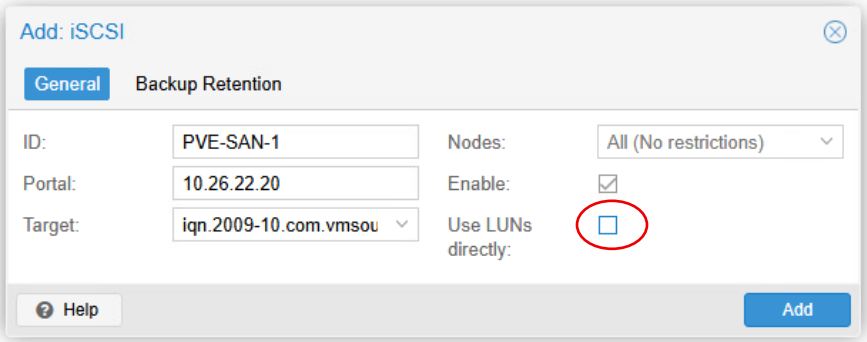
Step 2: Name: PVE-SAN-1

Step 3: Portal IP: 10.26.22.20

Step 4: Uncheck: Use LUNs directly

NOTE : This prevents users from creating a VM directly on the LUN, using the whole LUN for one VM

Step 5: Target iqn appears when discovered



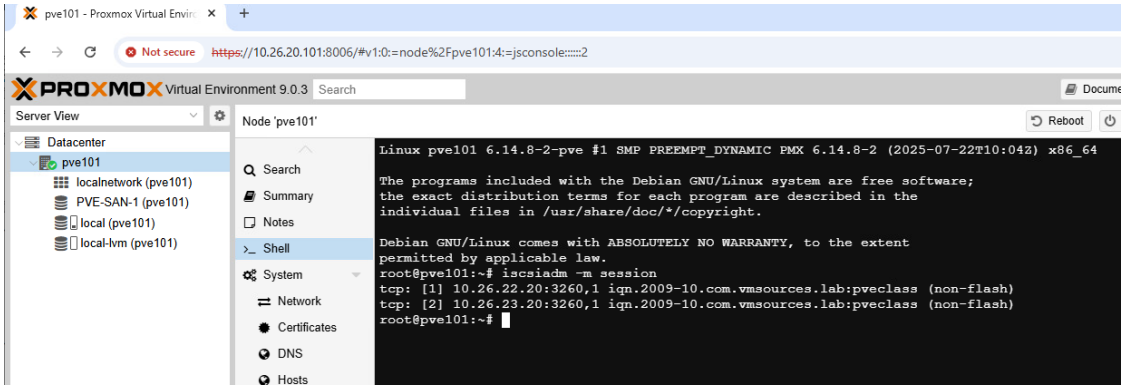
e.g.:

8. Now let's check our iSCSI connections.

Step 1: Go to: Shell

Command #1 run: iscsiadm -m session

Step 2:



NOTE : You should see both possible connections to our SAN

SBS LAB – (CLI/GUI) iSCSI Multipath with Targets on Same Network (Port Binding)

This SBS LAB requires a different networking configuration than what we have done so far in this process. In order for this lab to work the port corresponding to nic6 on your server will need to be changed from VLAN 23 ACCESS to VLAN 22 ACCESS and then refer back to ***SBS LAB –(GUI) iSCSI Interface Config for PVE with Jumbo Frames*** and configure nic5 with the IP: 10.26.22.XYZ and nic6 with the IP: 20.26.22.ABC. **This lab is included for reference in case someone is stuck in a situation with multiple targets on the same network**.

Step 1: Verify/edit our iSCSI configuration to make sure node.startup = manual

Command #1 run: vi /etc/iscsi/iscsid.conf

e.g.:

```
# open-iscsi can create a session and bind it to a NIC/HBA.
# To set this up see the example iface config file.

#*****************
# Startup settings
#*****************

# To request that the iscsi service scripts startup a session, use "automatic":
#node.startup = automatic
#
# To manually startup the session, use "manual". The default is manual.
node.startup = manual
```

NOTE : We do this because we do not want to automatically log-on to all portals reported during discovery

Command #1 run: systemctl restart iscsi.service

e.g.:

```
root@pve104:/# systemctl restart iscsi.service
root@pve104:/#
```

Step 2: Discover our iSCSI SAN

Command #1 run: iscsiadm -m discovery -t sendtargets -p 10.26.22.30

e.g.:

```
root@pve104:~# iscsiadm -m discovery -t sendtargets -p 10.26.22.30
10.26.22.30:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.20.98:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.22.31:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
root@pve104:~#
```

NOTE : iSCSI discovery reports what the SAN says is available, regardless of the relevance of some of the paths/targets to your use case. Here you see three potential paths/targets on our iSCSI SAN. 10.26.22.31 and 10.26.22.31 are relevant while 10.26.20.98 is the management IP of the SAN and not reachable from our iSCSI interfaces.

Step 3: Define your iSCSI interfaces

Command #1 run: iscsiadm -m iface -I iscsi1 -o new

Command #2 run: iscsiadm -m iface -I iscsi2 -o new

e.g.:

```
root@pve104:~# iscsiadm -m iface -I iscsi1 -o new
iscsi1 updated.
New interface iscsi1 added
root@pve104:~# iscsiadm -m iface -I iscsi2 -o new
iscsi2 updated.
New interface iscsi2 added
root@pve104:~#
```

NOTE : . In this step, we are merely assigning a name “iscsi1” and “iscsi2”

Step 4: Bind each named iSCSI interface to a network connection on PVE

Command #1 run: iscsiadm -m iface -I iscsi1 -o update -n iface.net_ifacename -v nic5

Command #2 run: iscsiadm -m iface -I iscsi2 -o update -n iface.net_ifacename -v nic6

e.g.:

```
root@pve104:~# iscsiadm -m iface -I iscsi1 -o update -n iface.net_ifacename -v nic5
iscsi1 updated.
root@pve104:~# iscsiadm -m iface -I iscsi2 -o update -n iface.net_ifacename -v nic6
iscsi2 updated.
root@pve104:~#
```

Step 5: Now that we are bound, we need to run iSCSI discovery again

Command #1 run: iscsiadm -m discovery -t sendtargets -p 10.26.22.30

e.g.:

```
root@pve104:~# iscsiadm -m discovery -t sendtargets -p 10.26.22.30
10.26.22.30:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.20.98:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.22.31:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.22.30:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.20.98:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
10.26.22.31:3260,1 iqn.2010-01.com.vmsources:tsn.803a6aff2c38
root@pve104:~#
```

NOTE : Now we see two instances for each reported path/target because we are now using two interfaces for iSCSI

Step 6: Log each appropriate iSCSI interface explicitly in to it's respective/desired target

Command #1 run: iscsiadm -m node -T iqn.2010-01.com.vmsources:tsn.803a6aff2c38 -p 10.26.22.30 -I iscsi1 --login

Command #2 run: iscsiadm -m node -T iqn.2010-01.com.vmsources:tsn.803a6aff2c38 -p 10.26.22.31 -I iscsi2 –login

e.g.:

```
root@pve104:~# iscsiadm -m node -T iqn.2010-01.com.vmsources:tsn.803a6aff2c38 -p 10.26.22.30 -I iscsi1 --login
Login to [iface: iscsi1, target: iqn.2010-01.com.vmsources:tsn.803a6aff2c38, portal: 10.26.22.30,3260] successful.
root@pve104:~# iscsiadm -m node -T iqn.2010-01.com.vmsources:tsn.803a6aff2c38 -p 10.26.22.31 -I iscsi2 --login
Login to [iface: iscsi2, target: iqn.2010-01.com.vmsources:tsn.803a6aff2c38, portal: 10.26.22.31,3260] successful.
root@pve104:~#
```

NOTE : Some SANs present many LUNs on one target (more common) and some present an unique target per LUN. If you saw many targets per IP in discovery (above), you would repeat this procedure for each target/IP listed in discovery.

Step 7: Set which portals will login automatically

Command #1 run: cd /var/lib/iscsi/nodes

Command #2 run: ls

e.g.:

```
root@pve104:/# cd /var/lib/iscsi/nodes/
root@pve104:/var/lib/iscsi/nodes# ls
iqn.2009-10.com.vmsources.lab:pveclass  iqn.2010-01.com.vmsources:tsn.803a6aff2c38
root@pve104:/var/lib/iscsi/nodes#
```

NOTE : Here you see any/all discovered iSCSI SAN IQN identifiers

Step 8: Change directory to the SAN/IQN we want to connect and list

Command #1 run: cd iqn.2010-01.com.vmsources:tsn.803a6aff2c38

Command #2 run: ls

e.g.:

```
root@pve104:/var/lib/iscsi/nodes# cd iqn.2010-01.com.vmsources:tsn.803a6aff2c38
root@pve104:/var/lib/iscsi/nodes/iqn.2010-01.com.vmsources:tsn.803a6aff2c38# ls
10.26.20.98,3260,1  10.26.22.30,3260,1  10.26.22.31,3260,1
root@pve104:/var/lib/iscsi/nodes/iqn.2010-01.com.vmsources:tsn.803a6aff2c38#
```

NOTE : Now you see all of the discovered portals on the SAN by IP and port

Step 9: Change to the irrelevant management IP/port

Command #1 run: cd 10.26.20.98,3260,1

Command #2 run: ls

e.g.:

```
root@pve104:/var/lib/iscsi/nodes/iqn.2010-01.com.vmsources:tsn.803a6aff2c38# cd 10.26.20.98,3260,1
root@pve104:/var/lib/iscsi/nodes/iqn.2010-01.com.vmsources:tsn.803a6aff2c38/10.26.20.98,3260,1# ls
iscsi1  iscsi2
root@pve104:/var/lib/iscsi/nodes/iqn.2010-01.com.vmsources:tsn.803a6aff2c38/10.26.20.98,3260,1#
```

Step 10: Now we will verify that node.startup is set to manual for both iscsi1 and iscsi2

Command #1 run: vi iscsi1

e.g.:

```
# BEGIN RECORD 2.1.11
node.name = iqn.2010-01.com.vmsources:tsn.803a6aff2c38
node.tpgt = 1
node.startup = manual
node.leading_login = No
iface.iscsi_ifacename = iscsi1
iface.net_ifacename = nic5
```

Command #1 run: vi iscsi2

e.g.:

```
# BEGIN RECORD 2.1.11
node.name = iqn.2010-01.com.vmsources:tsn.803a6aff2c38
node.tpgt = 1
node.startup = manual
node.leading_login = No
iface.iscsi_ifacename = iscsi2
```

NOTE : Make sure node.startup is set to manual (which it should be if you set is that way in /etx/iscsi/iscsid.conf and restarted the service)

Step 11: Now change to the first relevant IP/port using a relative path (../)

Command #1 run: cd ../10.26.22.30,3260,1

Command #2 run: vi iscsi1

e.g.:

```
# BEGIN RECORD 2.1.11
node.name = iqn.2010-01.com.vmsources:tsn.803a6aff2c38
node.tpgt = 1
node.startup = automatic
node.leading_login = No
iface.iscsi_ifacename = iscsi1
iface.net_ifacename = nic5
```

NOTE : For the target IP 10.26.22.30, we want the iscsi1 interface (10.26.22.XYZ) to connect automatically) so set node.startup to automatic and save (:wq)

Command #1 run: vi iscsi2

e.g.:

```
# BEGIN RECORD 2.1.11
node.name = iqn.2010-01.com.vmsources:tsn.803a6aff2c38
node.tpgt = 1
node.startup = manual
node.leading_login = No
iface.iscsi_ifacename = iscsi2
iface.net_ifacename = nic6
iface.prefix_len = 0
iface.transport_name = tcp
```

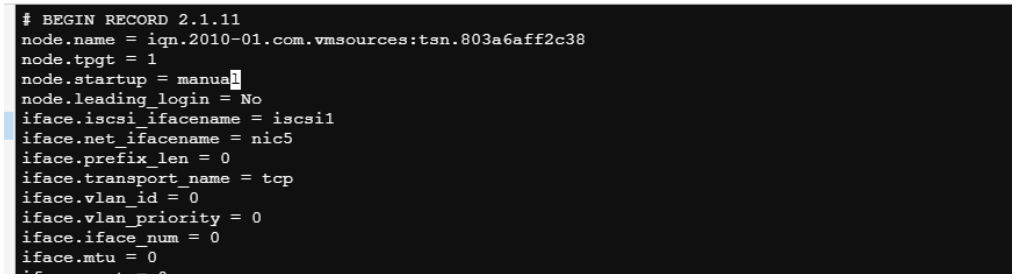
NOTE : For the target IP 10.26.22.30, we DO NOT want the iscsi2 interface (10.26.22.ABC) to connect automatically) so set node.startup to manual and save (:wq)

Step 12: Now change to the next relevant IP/port using a relative path (../)

Command #1 run: cd ../10.26.22.31,3260,1

Command #2 run: vi iscsi1

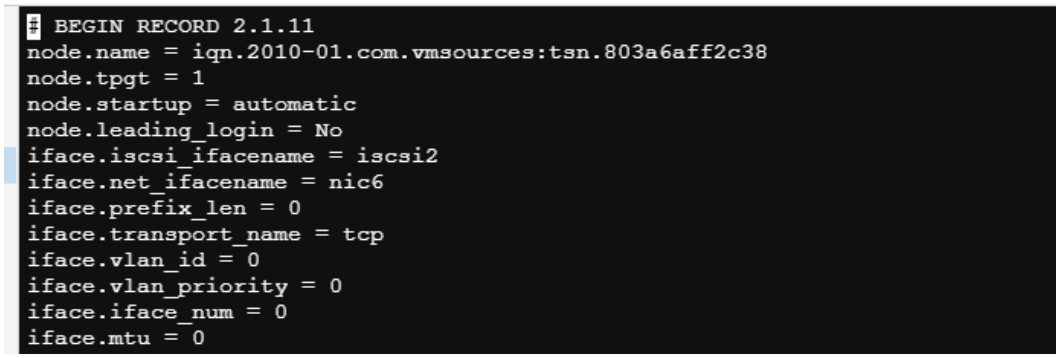
e.g.:



NOTE : For the target IP 10.26.22.30, we DO NOT want the iscsi1 interface (10.26.22.XYZ) to connect automatically) so set node.startup to manual and save (:wq)

Command #1 run: vi iscsi2

e.g.:

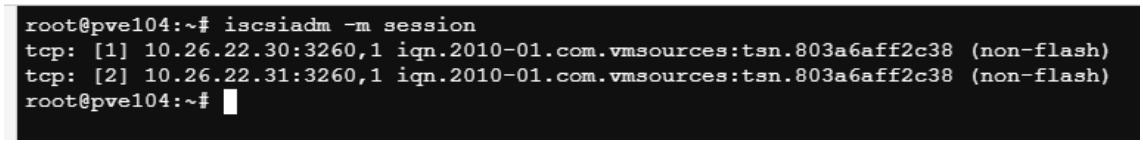


NOTE : For the target IP 10.26.22.31, we want the iscsi2 interface (10.26.22.ABC) to connect automatically) so set node.startup to automatic and save (:wq)

Step 13: Now check iSCSI sessions

Command #1 run: Iscsiadm -m session

e.g.:

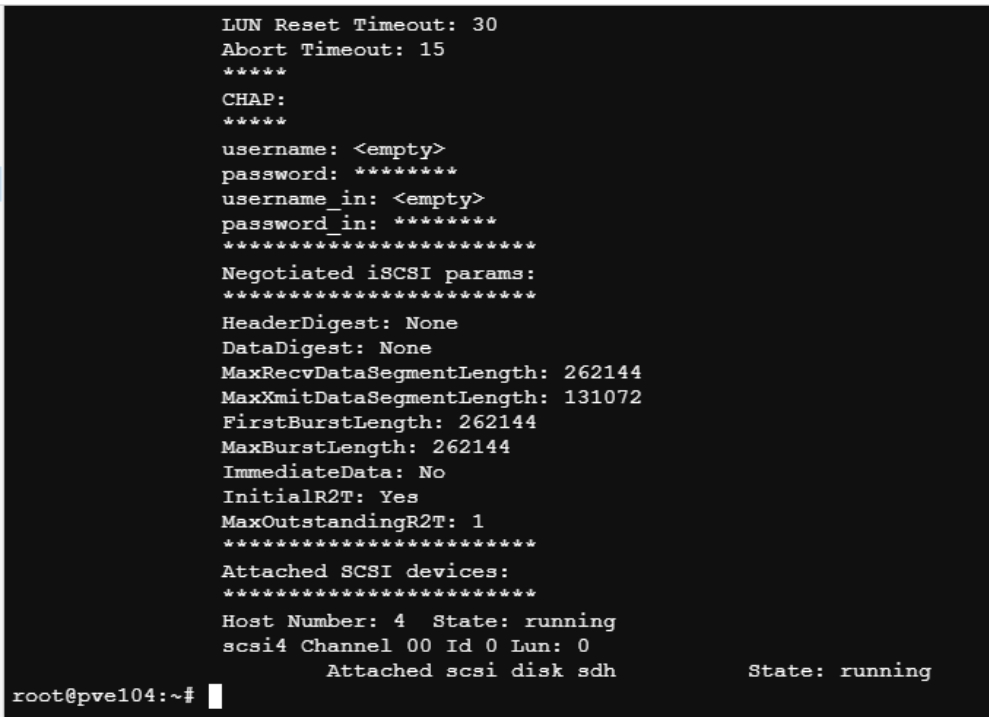


NOTE : There should be two active sessions, as intended, one for each iSCSI interface

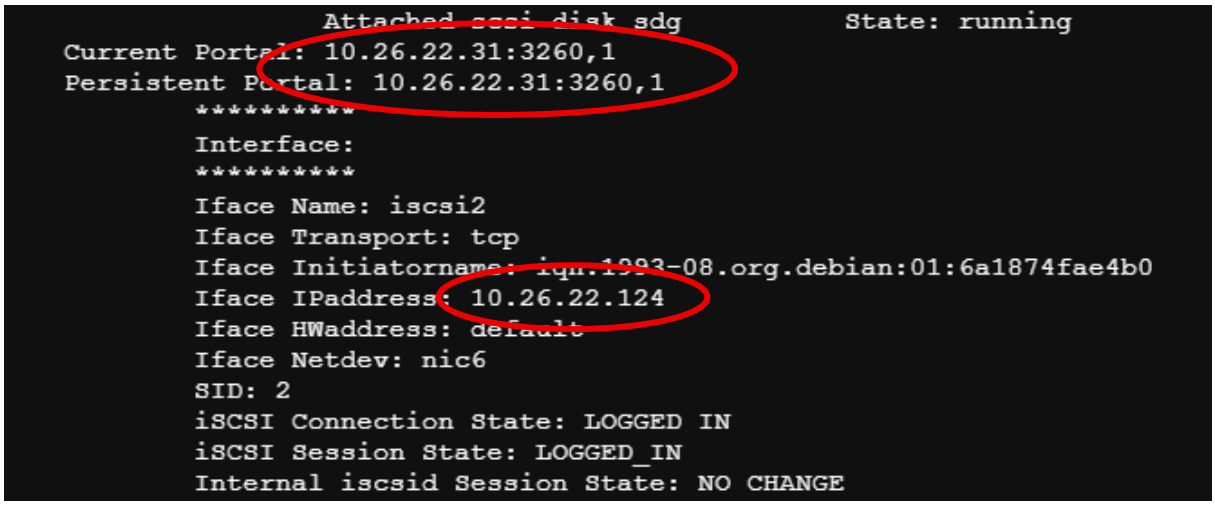
Step 14: Check iSCSI bindings

Command #1 run: iscsiadm -m session -P 3

e.g.:



e.g.:



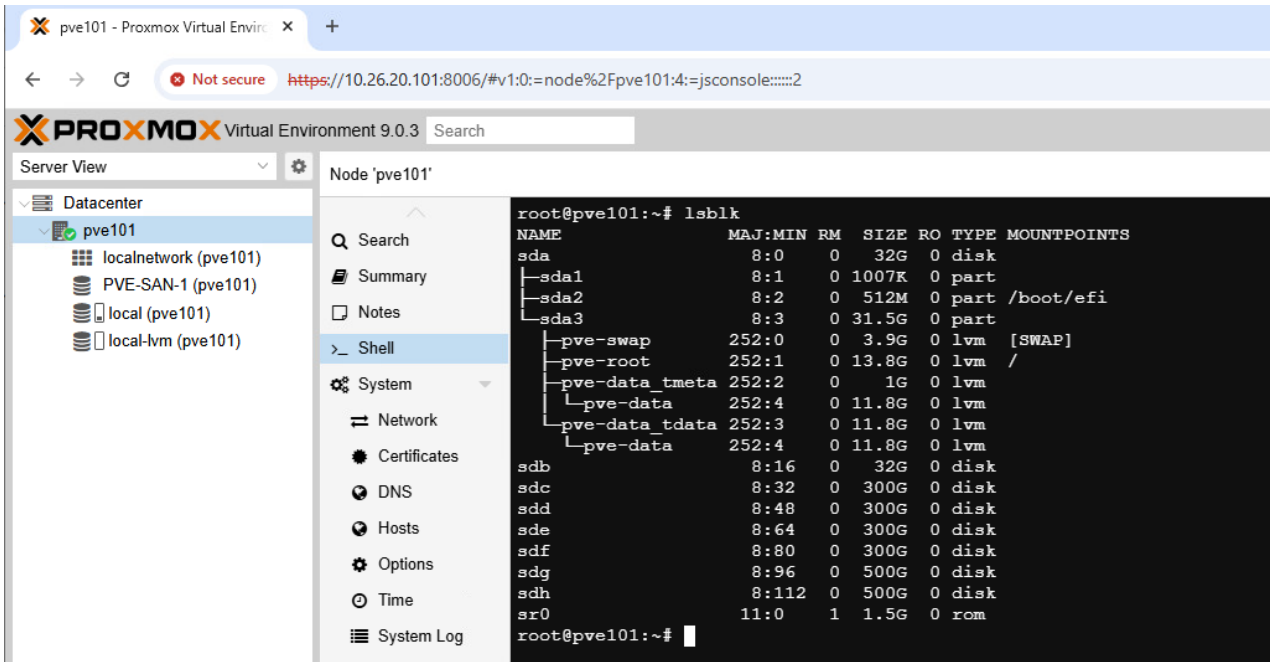
NOTE : You should see each defined iSCSI interface connected to one and only one SAN IP

SBS LAB – (CLI) Apply multipath to storage

Step 1: Identify volumes as seen

Command #1 run: lsblk

e.g.:



NOTE : What you see here are any number of local disks on the PVE server and multiple disks presented from the SAN which may or may not be unique or simply separate paths to the same disk

Step 2: Let's dig a little deeper to make sure.

Command #1 run: lsblk -o NAME,KNAME,TYPE,HCTL,SIZE,MODEL,SERIAL

e.g.:

```
root@pve101:~# lsblk -o NAME,KNAME,TYPE,HCTL,SIZE,MODEL,SERIAL
NAME                KNAME TYPE HCTL         SIZE MODEL                          SERIAL
sda                 sda   disk 2:0:0:0       32G Virtual disk
├─sda1              sda1  part              1007K
├─sda2              sda2  part               512M
└─sda3              sda3  part              31.5G
  ├─pve-swap        dm-0  lvm               3.9G
  ├─pve-root        dm-1  lvm              13.8G
  ├─pve-data_tmeta  dm-2  lvm                 1G
  │ └─pve-data      dm-4  lvm              11.8G
  └─pve-data_tdata  dm-3  lvm              11.8G
    └─pve-data      dm-4  lvm              11.8G
sdb                 sdb   disk 2:0:1:0       32G Virtual disk
sdc                 sdc   disk 2:0:2:0      300G Virtual disk
sdd                 sdd   disk 2:0:4:0      300G Virtual disk
sde                 sde   disk 2:0:3:0      300G Virtual disk
sdf                 sdf   disk 2:0:5:0      300G Virtual disk
sdg                 sdg   disk 3:0:0:101    500G iSCSI Disk                     2e41368b0786686
sdh                 sdh   disk 4:0:0:101    500G iSCSI Disk                     2e41368b0786686
sr0                 sr0   rom  0:0:0:0      1.5G VMware Virtual IDE CDROM Drive 00000000000000000001
root@pve101:~#
```

NOTE : You can identify SAN LUNs because they have a serial number. Pay no attention to the 'iSCSI Disk' information as that is data provided by the SAN and could be "uncle Bob" if they wanted it to be. Each unique LUN from the SAN has an unique serial number. If you see that PVE identifies multiple "disks"(/dev/sdX) by the same serial number, those represent multiple pathways to the same LUN on the SAN

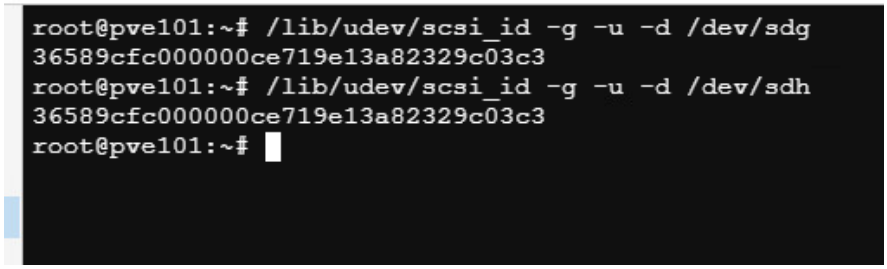
NOTE : Make note of which /dev/sdX disks have a serial number

Step 3: Now determine the WWID of each SAN disk

Command #1 run: /lib/udev/scsi_id -g -u -d /dev/sdX

NOTE : Where X is disk identified by a serial number. Run this once for all of the available disk identified by a serial number.

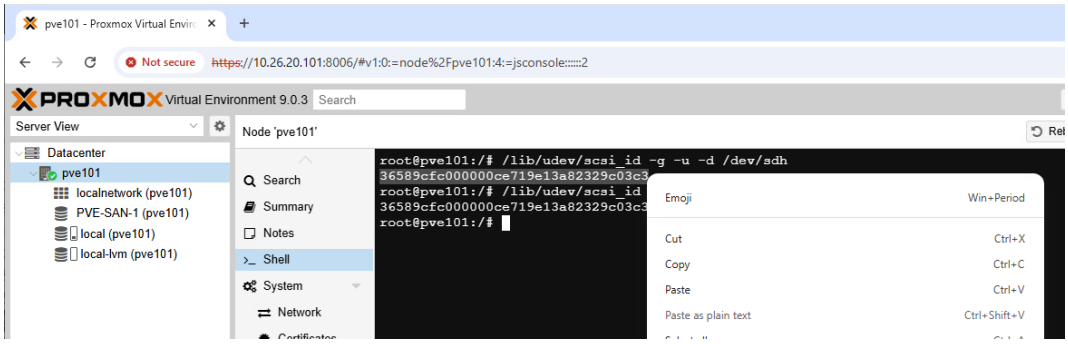
e.g.:



NOTE : These numbers are the 'wwid' (World Wide Identification) of your iSCSI LUNs. Make note of each unique number in Notepad++, we will use them later

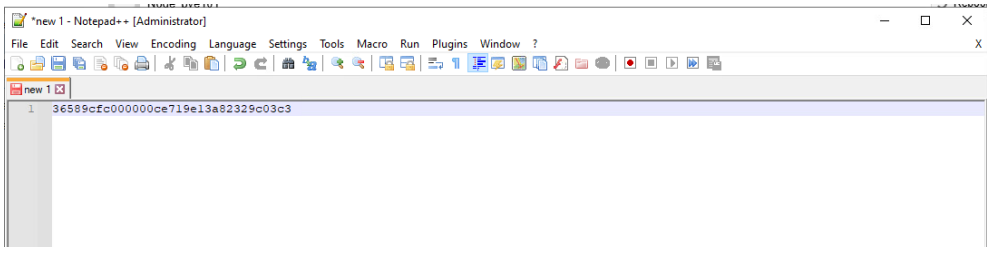
Step 4: Copy the all the unique wwid's

e.g.:

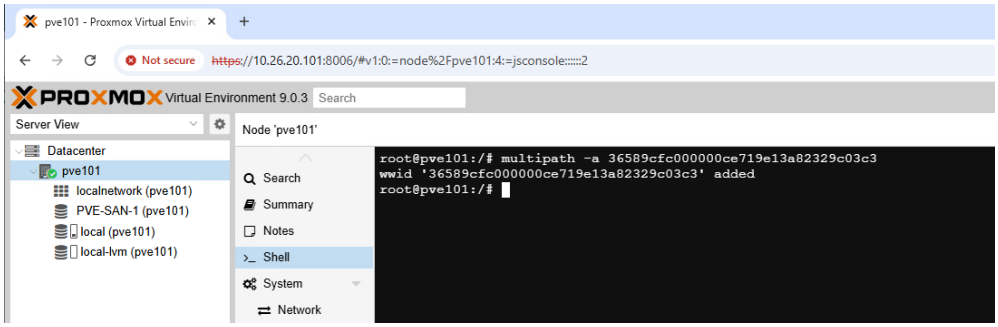


Step 5: Paste it in Notepad++

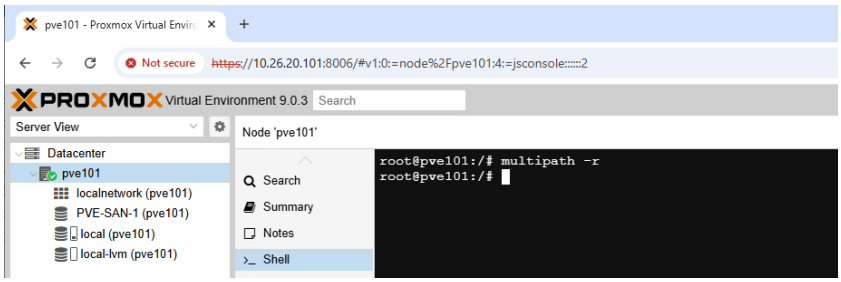
e.g.:



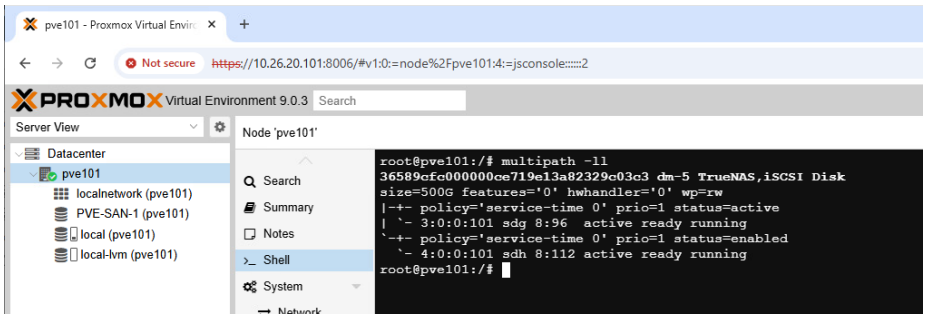
13. Now we need to add all of the wwid's to iSCSI multipath.
    Command #1 run: multipath -a wwid for each unique wwid you have recorded
    e.g.:



13. Activate these wwid additions.
    Command #1 run: multipath -r
    e.g.:



14. Now verify your configuration.
    Command #1 run: multipath -ll
    e.g.:



    NOTE : You may also see 'pending' at this point

IN PRODUCTION: A LUN (Logical Unit Number) is presented by the SAN, a disk is something like /dev/sdX which is identified by PVE.

SBS LAB – (GUI) Create LVM Volume

In this lab we will create an LVM volume. **It is important NOT to use LVM-Thin in multi-node environments** connection to shared storage as LVM is not able to handle thinly-provisioned volumes in shared environments.

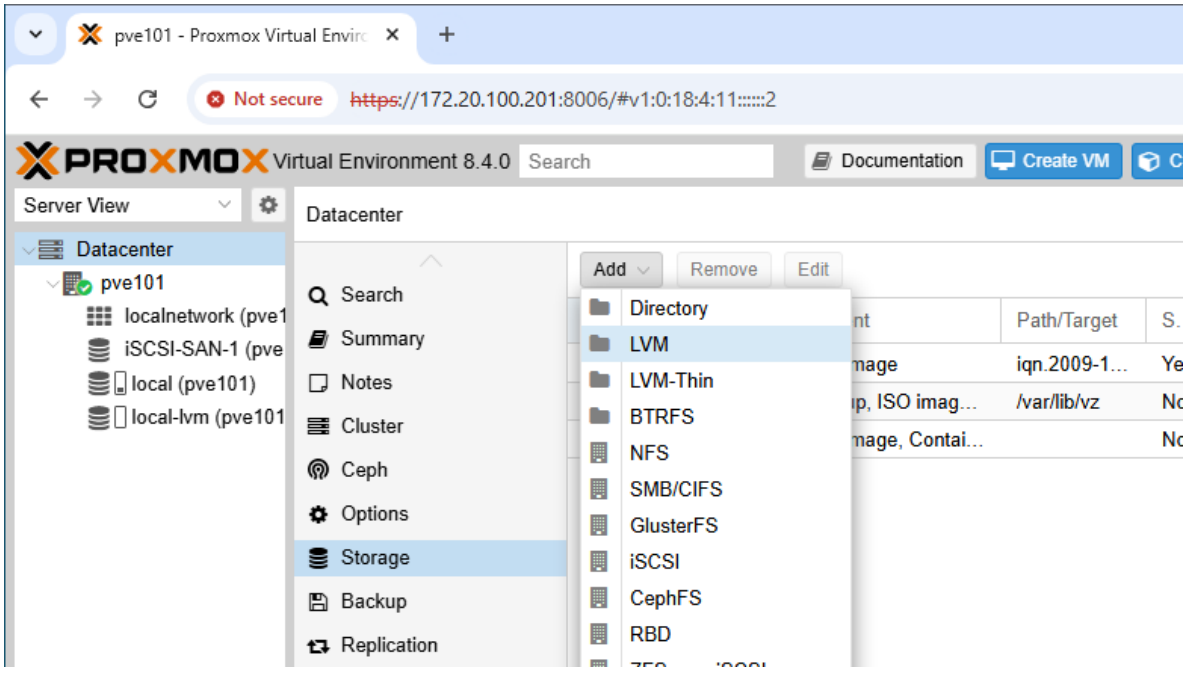
Variables for this lab (or other as appropriate for your environment):

- ID: PVE-LVM-101 (yes, we are all adding the same LUN 101 for now)
- Base storage: PVE-SAN-1
- Base volume: CH 00 ID 0 LUN 101 (yes, we are all adding the same LUN 101 for now)
- Volume Group: PVE-VG-1

> IN PRODUCTION:
>
> You will only Create LVM Volumes on the cluster first node and those configs will be copied to any other nodes that join the cluster.

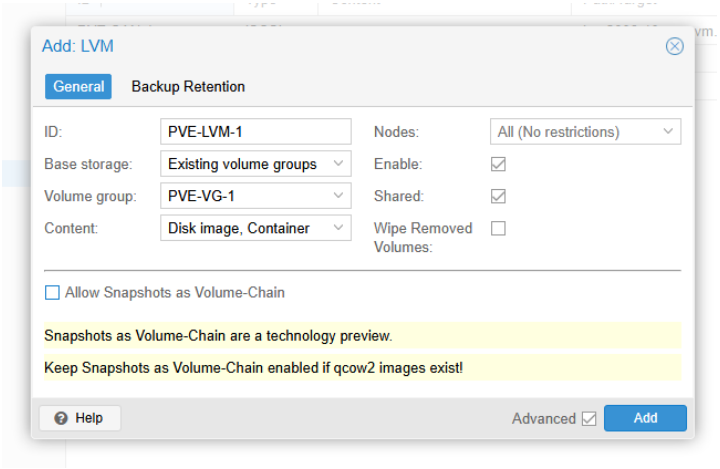
Step 1: Now go to: Datacenter > Storage > Add > LVM

e.g.:



Step 2: ID: PVE-LVM-1
Step 3: Choose the SAN (Base storage)
Step 4: Base Volume: CH 00 ID 0 LUN 101
Step 5: Volume group: PVE-VG-1
Step 6: Content: Disk image, Container
Step 7: Enabled: Checked
Step 8: Shared: Checked
Step 9: Allow Snapshots as Volume-Chain: Checked
Step 10: Select Add

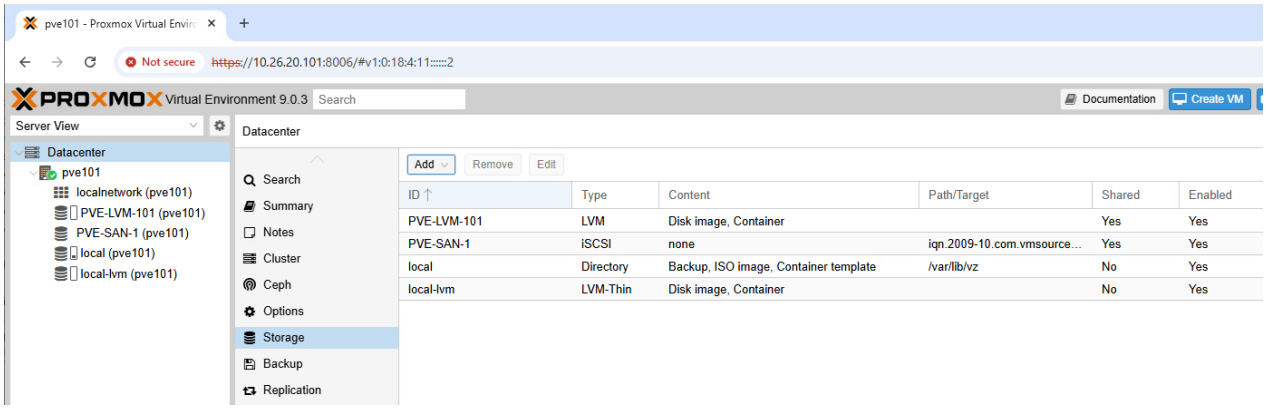
e.g.:



IN PRODUCTION:

Selecting Snapshots as Volume-Chain will decrease performance for the whole volume by 30-90% (https://kb.blockbridge.com/technote/proxmox-qcow-snapshots-on-lvm/)

e.g.:



SBS LAB – (CLI) PVE Storage /SAN Diagnostics

Step 1: Open a shell to your PVE node

Command #1 run: df -h

e.g.:

```
root@pve102:~# df -h
Filesystem            Size  Used Avail Use% Mounted on
udev                   32G     0   32G   0% /dev
tmpfs                 6.3G  1.5M  6.3G   1% /run
/dev/mapper/pve-root   14G  4.1G  8.8G  32% /
tmpfs                  32G   69M   32G   1% /dev/shm
efivarfs              256K   38K  214K  15% /sys/firmware/efi/efivars
tmpfs                 5.0M     0  5.0M   0% /run/lock
tmpfs                 1.0M     0  1.0M   0% /run/credentials/systemd-journald.servi
ce
tmpfs                  32G     0   32G   0% /tmp
/dev/sda2             511M  8.8M  503M   2% /boot/efi
/dev/fuse             128M   28K  128M   1% /etc/pve
tmpfs                 1.0M     0  1.0M   0% /run/credentials/getty@tty1.service
tmpfs                  32G   80K   32G   1% /var/lib/ceph/osd/ceph-0
tmpfs                  32G   80K   32G   1% /var/lib/ceph/osd/ceph-1
tmpfs                  32G   80K   32G   1% /var/lib/ceph/osd/ceph-2
tmpfs                  32G   80K   32G   1% /var/lib/ceph/osd/ceph-3
tmpfs                 6.3G  4.0K  6.3G   1% /run/user/0
root@pve102:~#
```

NOTE : Shows the basic layout of the PVE installation

Command #1 run: fdisk -l

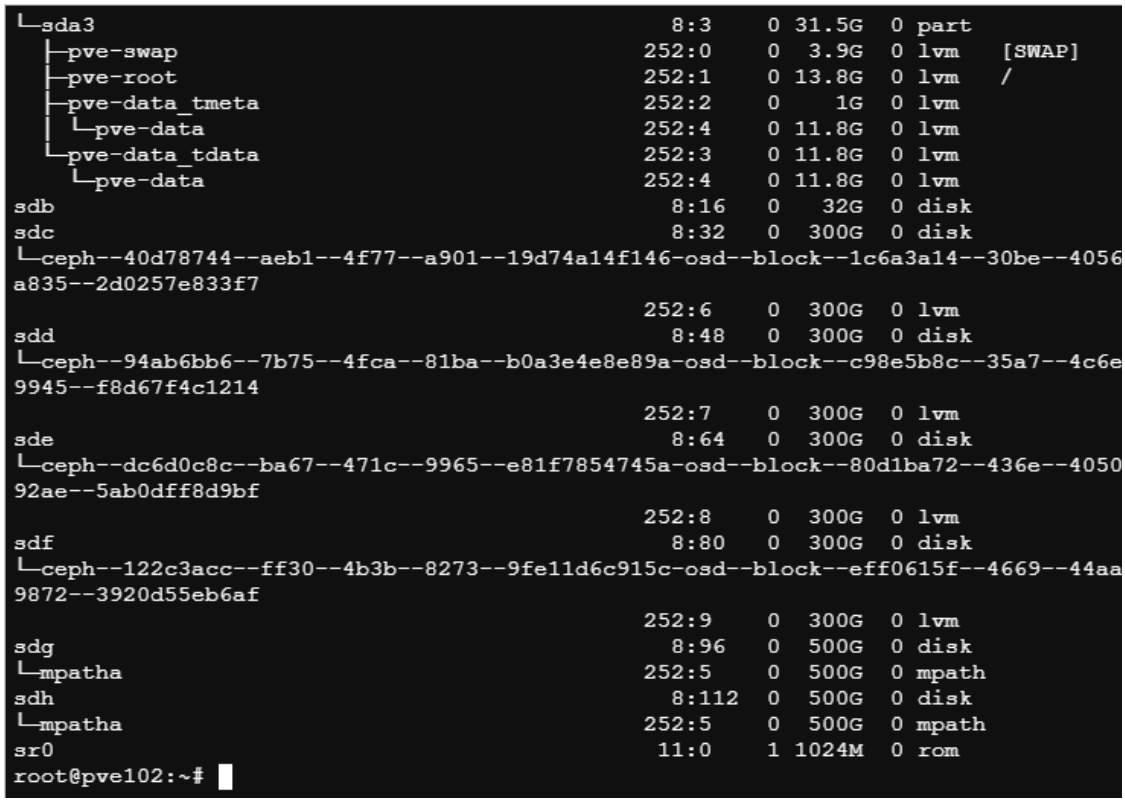
e.g.:

```
I/O size (minimum/optimal): 65536 bytes / 8388608 bytes


Disk /dev/mapper/ceph--40d78744--aeb1--4f77--a901--19d74a14f146-osd--block--1c6a3a1
4--30be--4056--a835--2d0257e833f7: 300 GiB, 322118352896 bytes, 629137408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ceph--94ab6bb6--7b75--4fca--81ba--b0a3e4e8e89a-osd--block--c98e5b8
c--35a7--4c6e--9945--f8d67f4c1214: 300 GiB, 322118352896 bytes, 629137408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ceph--dc6d0c8c--ba67--471c--9965--e81f7854745a-osd--block--80d1ba7
2--436e--4050--92ae--5ab0dff8d9bf: 300 GiB, 322118352896 bytes, 629137408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes


Disk /dev/mapper/ceph--122c3acc--ff30--4b3b--8273--9fe11d6c915c-osd--block--eff0615
f--4669--44aa--9872--3920d55eb6af: 300 GiB, 322118352896 bytes, 629137408 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
root@pve102:~#
```

NOTE : Shows all of the disks mounted to PVE

Command #1 run: lsblk

e.g.:



NOTE : Shows the layout of block devices mounted to PVE

Command #1 run: cat /etc/pve/storage.cfg

e.g.:

```
root@pve102:~# cat /etc/pve/storage.cfg
dir: local
        path /var/lib/vz
        content backup,vztmpl,iso

lvmthin: local-lvm
        thinpool data
        vgname pve
        content images,rootdir

iscsi: PVE-SAN-1
        portal 10.26.22.20
        target iqn.2009-10.com.vmsources.lab:pveclass
        content none

lvm: PVE-LVM-101
        vgname PVE-VG-1
        base PVE-SAN-1:0.0.101.scsi-36589cfc000000ce719e13a82329c03c3
        content rootdir,images
        saferemove 0
        shared 0
        snapshot-as-volume-chain 1

root@pve102:~#
```

NOTE : Shows storage configuration

Command #1 run: pvesm status

e.g.:

```
root@pve102:~# pvesm status
Name               Type     Status           Total            Used       Available
        %
PVE-LVM-101         lvm     active       524279808        33566720       490713088
    6.40%
PVE-SAN-1         iscsi     active               0               0               0
    0.00%
local               dir     active        14145416         4355524         9049540
   30.79%
local-lvm       lvmthin     active        12378112               0        12378112
    0.00%
root@pve102:~#
```

NOTE : Shows type/total/used/available on block storage

Command #1 run: pvesm list PVE-SAN-1

e.g.:

```
root@pve102:~# pvesm list PVE-SAN-1
Volid                                                 Format  Type
Size VMID
PVE-SAN-1:0.0.101.scsi-36589cfc000000ce719e13a82329c03c3 raw     images    53687097
7536
root@pve102:~#
```

Command #1 run: iscsiadm -m session

e.g.:

```
root@pve102:~# iscsiadm -m session
tcp: [1] 10.26.22.20:3260,1 iqn.2009-10.com.vmsources.lab:pveclass (non-flash)
tcp: [2] 10.26.23.20:3260,1 iqn.2009-10.com.vmsources.lab:pveclass (non-flash)
root@pve102:~#
```

Command #1 run: pvs

e.g.:

```
root@pve102:~# pvs
  PV                VG                                    Fmt  Attr PSize    P
Free
  /dev/mapper/mpatha PVE-VG-1                             lvm2 a--   499.99g 4
67.98g
  /dev/sda3         pve                                   lvm2 a--   <31.50g
 8.00m
  /dev/sdc          ceph-40d78744-aeb1-4f77-a901-19d74a14f146 lvm2 a--  <300.00g
    0
  /dev/sdd          ceph-94ab6bb6-7b75-4fca-81ba-b0a3e4e8e89a lvm2 a--  <300.00g
    0
  /dev/sde          ceph-dc6d0c8c-ba67-471c-9965-e81f7854745a lvm2 a--  <300.00g
    0
  /dev/sdf          ceph-122c3acc-ff30-4b3b-8273-9fe11d6c915c lvm2 a--  <300.00g
    0
root@pve102:~#
```

NOTE : Shows mount paths/size/free

Command #1 run: vgs

e.g.:

```
root@pve102:~# vgs
  VG                                        #PV #LV #SN Attr   VSize    VFree
  PVE-VG-1                                    1   2   0 wz--n-  499.99g 467.98g
  ceph-122c3acc-ff30-4b3b-8273-9fe11d6c915c   1   1   0 wz--n- <300.00g      0
  ceph-40d78744-aeb1-4f77-a901-19d74a14f146   1   1   0 wz--n- <300.00g      0
  ceph-94ab6bb6-7b75-4fca-81ba-b0a3e4e8e89a   1   1   0 wz--n- <300.00g      0
  ceph-dc6d0c8c-ba67-471c-9965-e81f7854745a   1   1   0 wz--n- <300.00g      0
  pve                                         1   3   0 wz--n-  <31.50g   8.00m
root@pve102:~#
```

NFS Storage for PVE

Network File Storage (NFS) is a widely used file-level means of storing data, including VMs and ISO Images for PVE. As a network protocol, NFS utilizes configured load-balancing and frame-size for the adapters/network(s) to which it is connected.

NFS is thin-provisioned by default (although you can specify pre-allocation = reservation for file size) and supports Snapshots.

> IN PRODUCTION:
>
> You will only perform NFS storage configuration on the cluster first node and those configs will be copied to any other nodes that join the cluster.

SBS LAB – (GUI) NFS for PVE

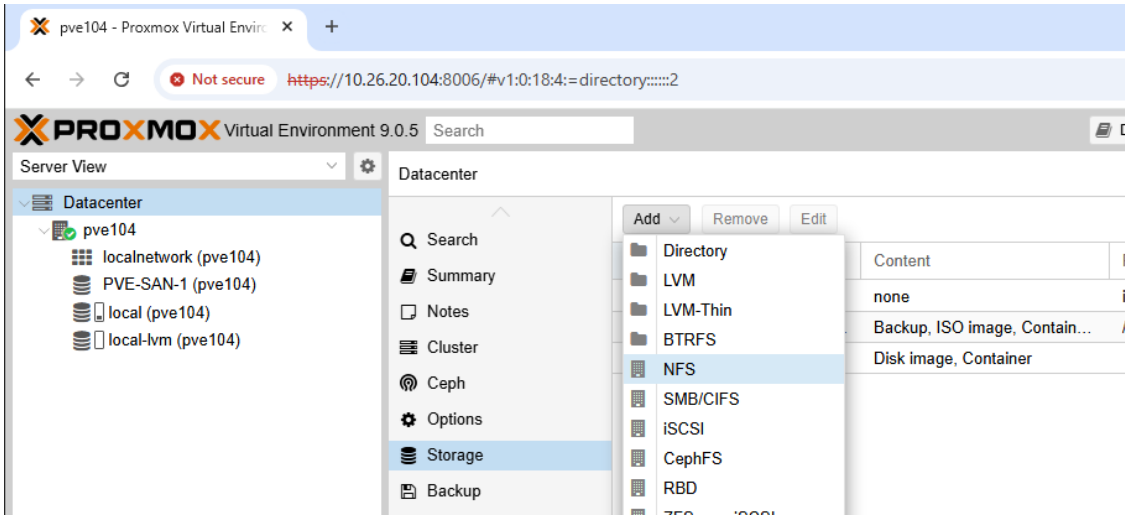
In this lab, we will connect to NFS storage presented on our PVE management VLAN 20 (10.26.20.0/24). The reason that we are not using one of our iSCSI networks, even though Jumbo Frames would benefit NFS), is that there would be no failover or load-balancing possible.

Hypothetically, given enough adapters, you could create a Bond on two or more Jumbo Frame capable adapters, set the optimum load-balancing method for your network, and utilize that as your NFS interface from PVE.

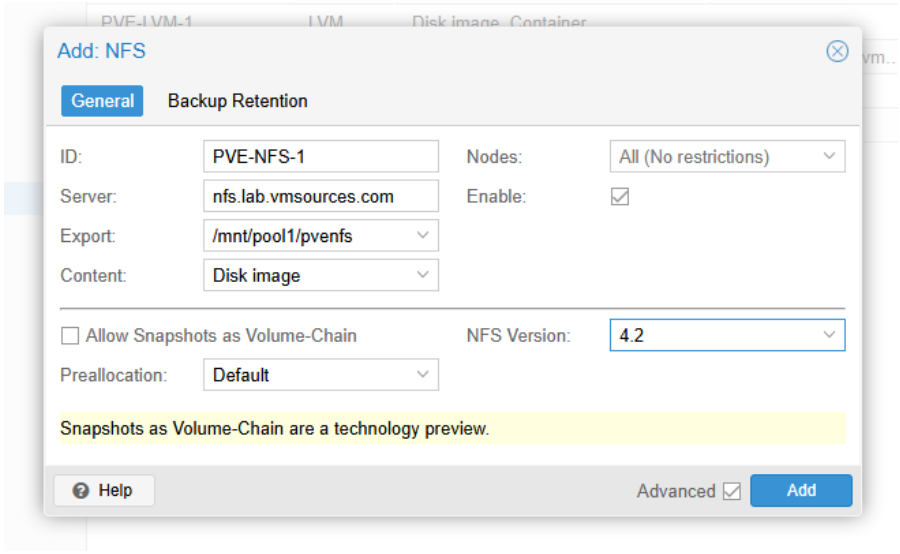
1. Configure NFS for PVE

   Step 1: Datacenter > Storage > Add > NFS

   e.g.:

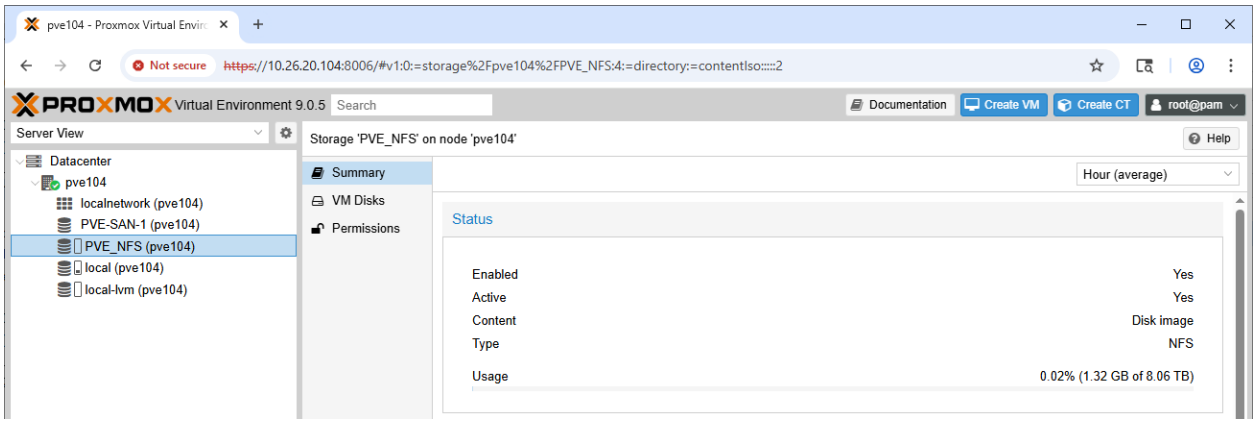


Step 2: ID: PVE-NFS-1
Step 3: Server: nfs.lab.vmsources.com
Step 4: Export: /mnt/pool1/pvenfs (should be automatically populated)
Step 5: Content: Disk Image
Step 6: NFS Version: 4.2 (Highest supported by your NFS Server))
e.g.:



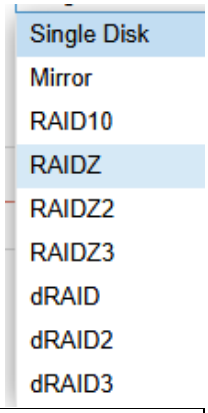
e.g.:

2. Complete and ready for use



e.g.:

ZFS Storage for PVE

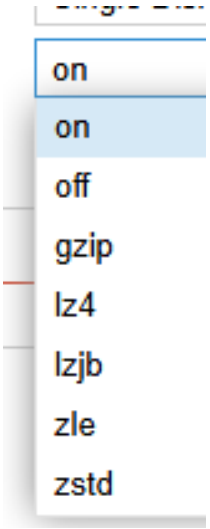


ZFS began life as open source Zettabyte File System from Sun Microsystems and eventually became closed source when Oracle Corporation acquired Sun. The original open source ZFS was forked around 2010, eventually becoming OpenZFS.

ZFS is designed around data integrity and scalability. Like most hyperconverged platforms (ZFS is not, technically HCI), ZFS needs to access disks individually (HBA mode) and not as part of a RAID set.

ZFS provides RAID-like availability through RAID-Z modes:

RAID Type	Parity / Redundancy	Disk Failure Tolerance	Performance
Mirror	Copy	50%	10
RAID10	Striped Mirror	50%	10
RAIDZ1	Single parity	N-1	6
RAIDZ2	Double parity	N-2	8
RAIDZ3	Triple parity	N-3	5
dRAID (Distributed RAID)	Single parity	N-1	8
dRAID2 (Distributed RAID)	Double parity	N-2	8
dRAID3 (Distributed RAID)	Triple parity	N-3	6



ZFS is also capable of compression:

Compression = on means lz4 compression, which is generally considered to be the most efficient overall.

- gzip offering better compression at the expense of CPU use and speed.
- lz4 (default) offers good mix of compression and minimum CPU use
- lzjb deprecated
- zle only compresses zeros
- zstd good compression, higher CPU use

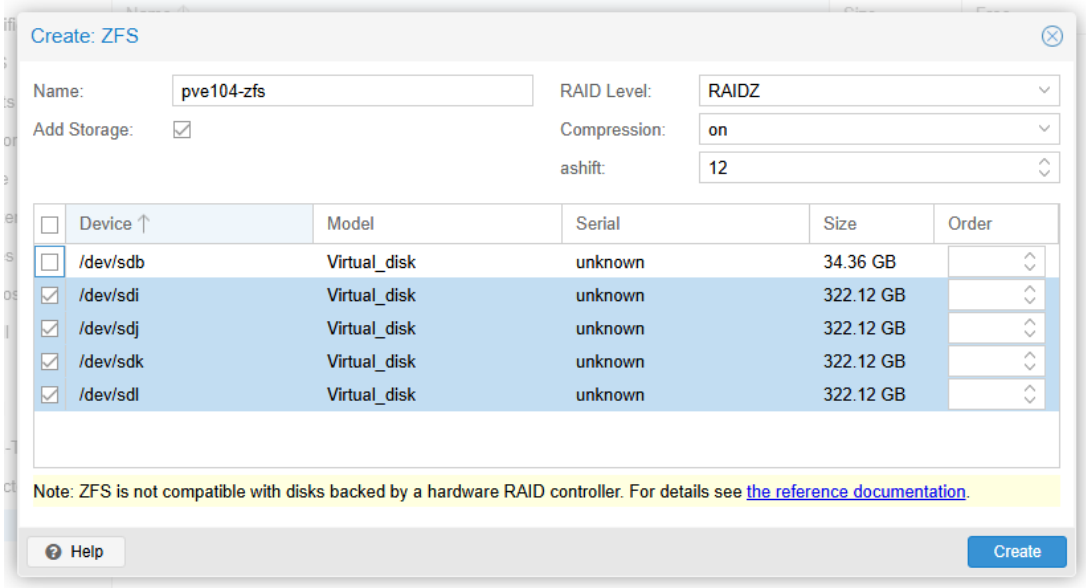
SBS LAB – (GUI) ZFS for PVE

In practice ZFS and Ceph are usually an either/or consideration as there will not be unlimited physical disks to deploy filesystems. This information is included for reference as we plan on deploying Ceph in subsequent steps.

1. Configure ZFS for PVE

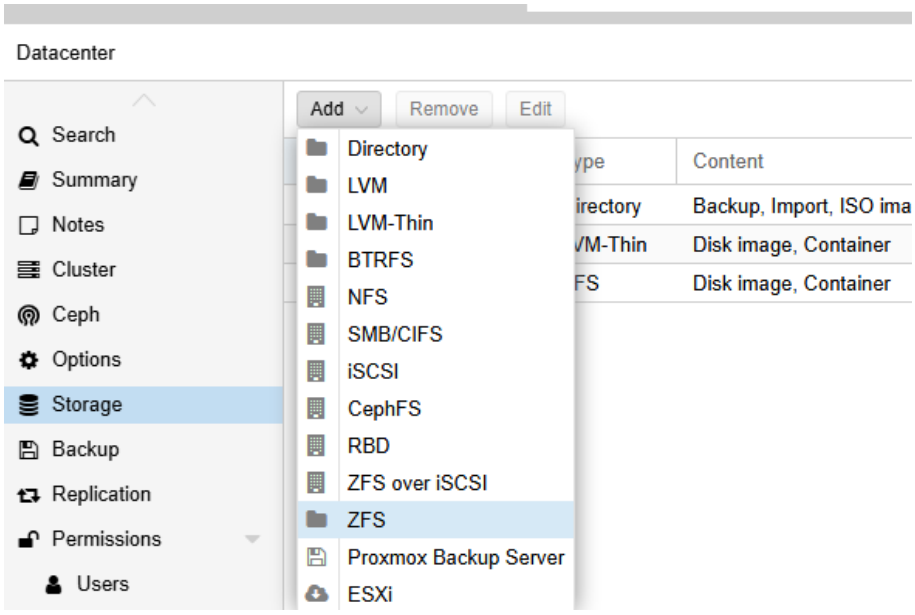
   Step 1: pveXYZ > ZFS > Create ZFS

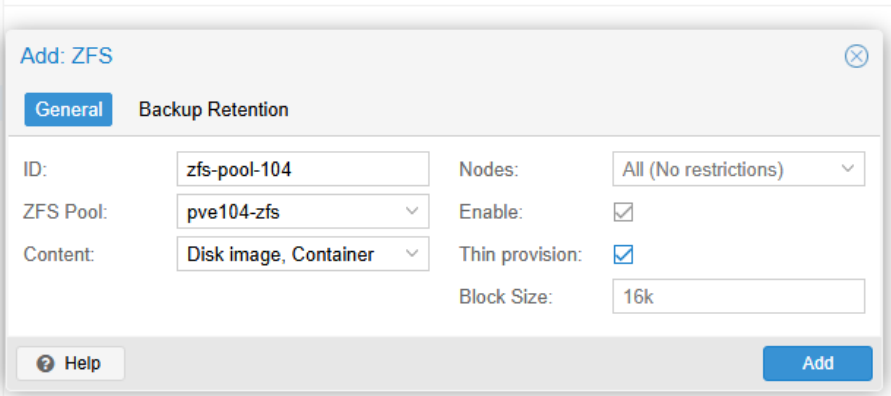
   e.g.:



   Step 2: Datacenter > Storage > Add > ZFS

   e.g.:




Add: ZFS
General
Backup Retention
ID:
zfs-pool-104
Nodes:
All (No restrictions)
ZFS Pool:
pve104-zfs
Enable:
Content:
Disk image, Container
Thin provision:
Block Size:
16k
Help
Add

e.g.:

Clustering PVE

PVE Clusters are very easy to create but you must keep in mind that the cluster mechanism called Corosync operate best with an odd number of nodes. Just like in any group environment, if there is an even number a tie is possible, creating a stalemate situation.

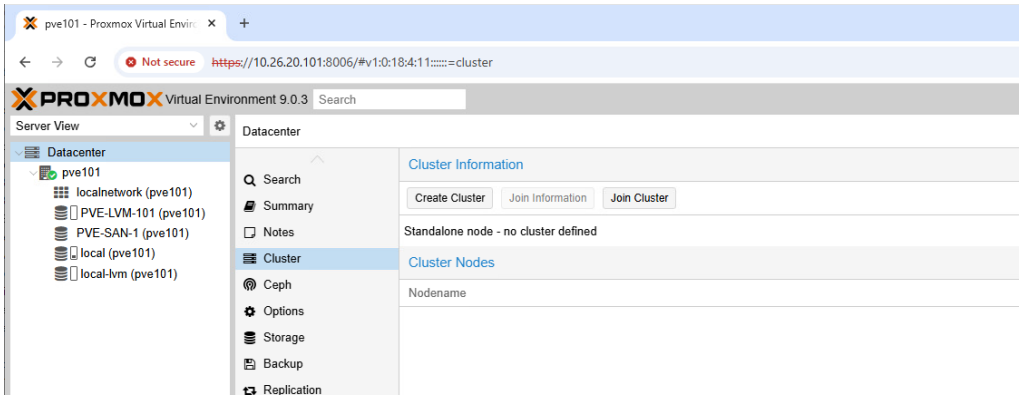
In Corosync clusters, if an even number of nodes is to be deployed, it is possible to deploy an external witness node (a 'Qdevice') on any system where Debian Linux can be installed.

Also, Corosync clusters do not always get along with network load-balancing mechanisms, so it is best to use a physical interface(s) for cluster communication.

SBS LAB – (GUI) Create Cluster on Primary node

Step 1: Click > Create Cluster

e.g.:



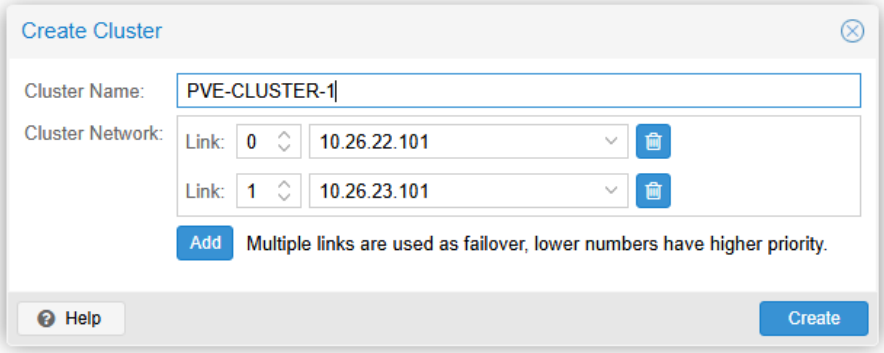
Step 2: Name the Cluster: PVE-CLUSTER-1

Step 3: Set Link 0 to the IP: 10.26.22.XYZ

e.g.:

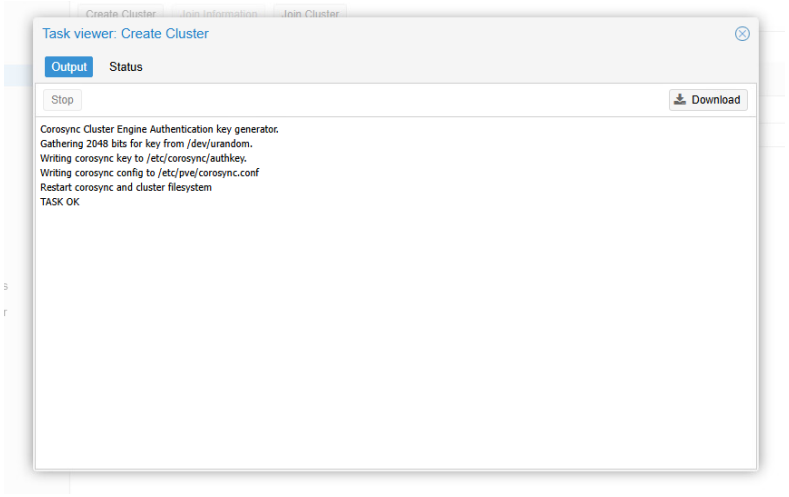
Step 4: Add

Step 5: Set the Link 1 to the IP: 10.26.23.XYZ



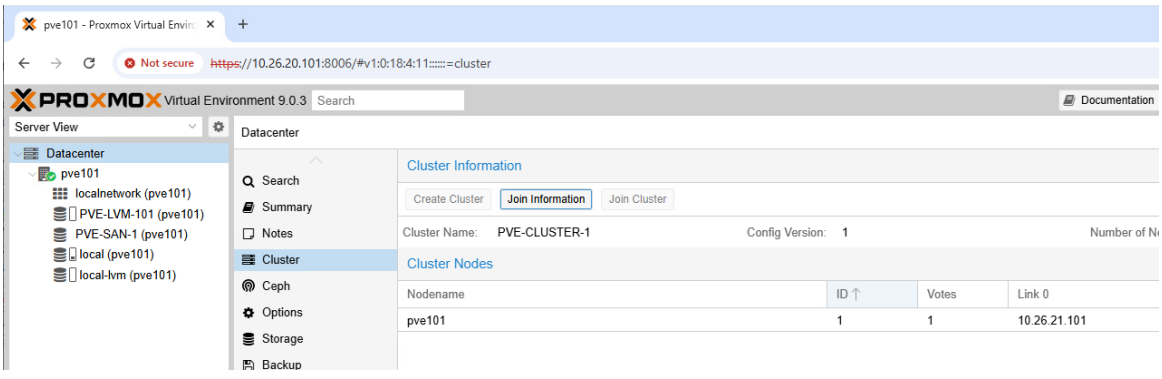
e.g.:

Step 6: Click > Create

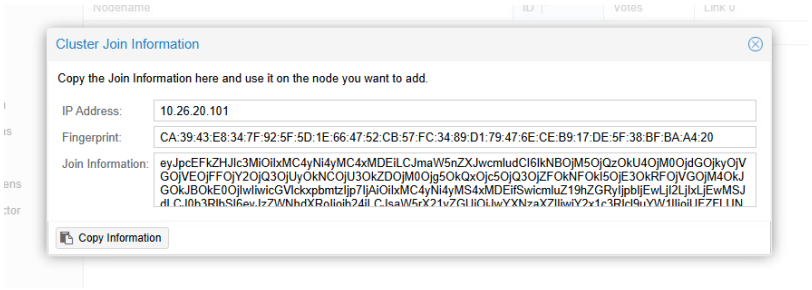


e.g.:

Step 7: Now click > Join Information



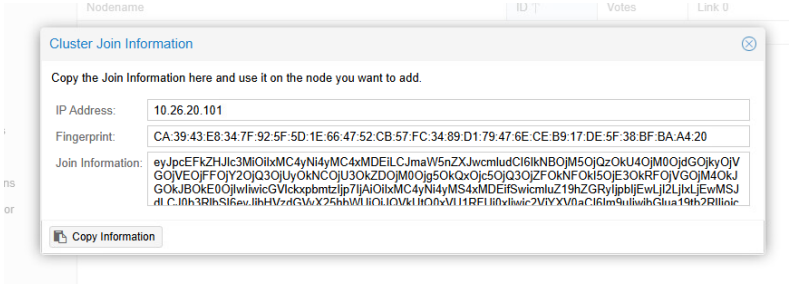
e.g.:



SBS LAB – (GUI) Joining PVE Nodes to Cluster

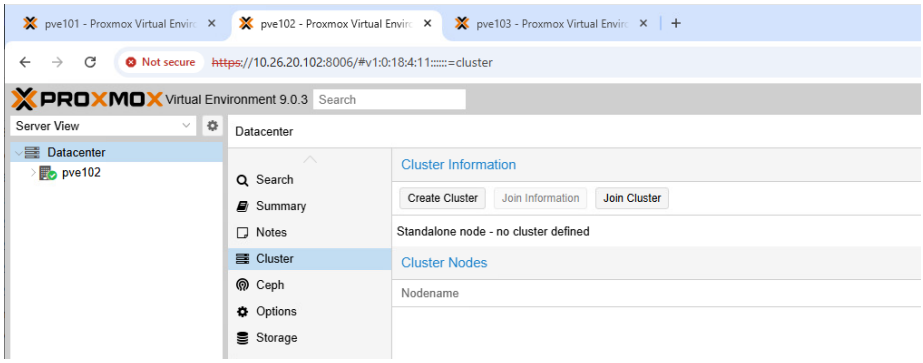
Step 1: Find and copy join information

e.g.:



Step 2: On your (or the next) PVE node, go to: Datacenter > Cluster > Join Cluster

e.g.:



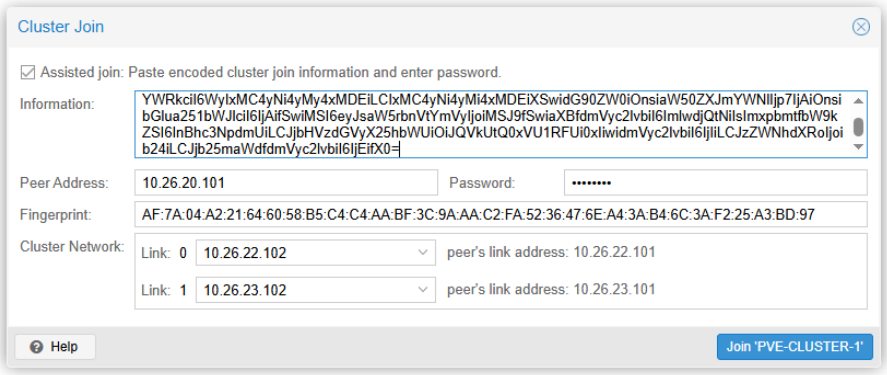
Step 3: Paste the join information

Step 4: Set Link 0 to the 10.26.22.XYZ for that node

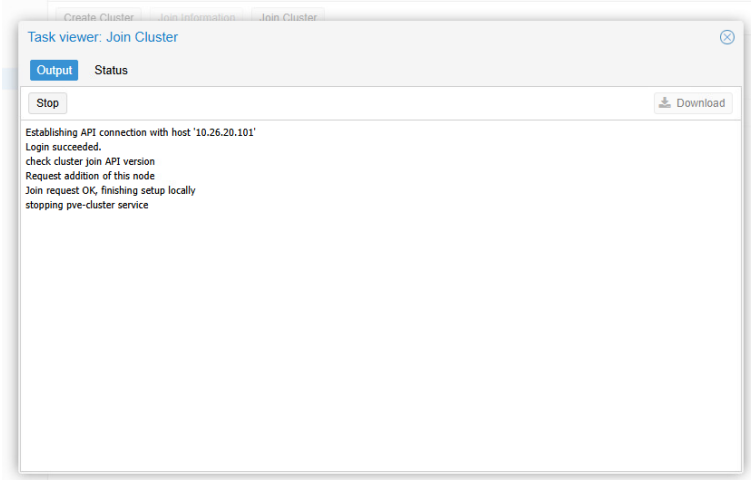
Step 5: Set Link 0 to the 10.26.23.XYZ for that node

Step 6: Provide root password of first cluster PVE node

Step 7: Join PVE Cluster



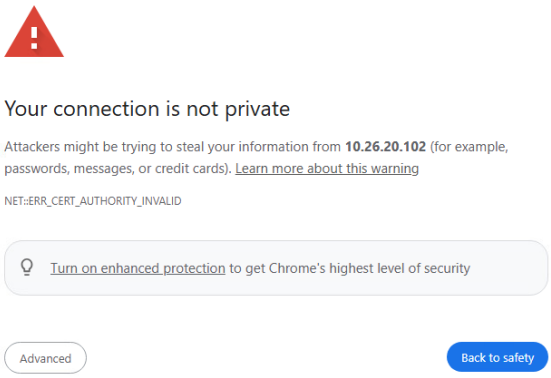
e.g.:



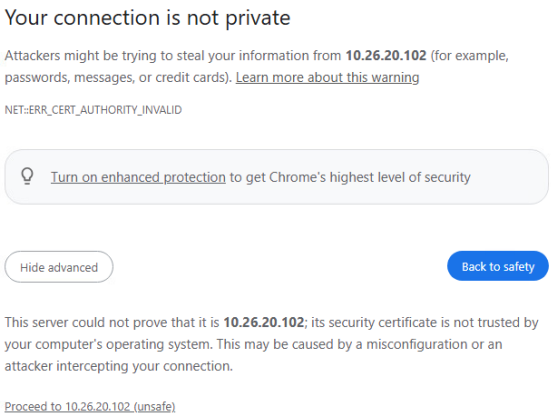
e.g.:

NOTE : Now your GUI will “freeze” here because your PVE node just got a new certificate

Step 8: Click: reload

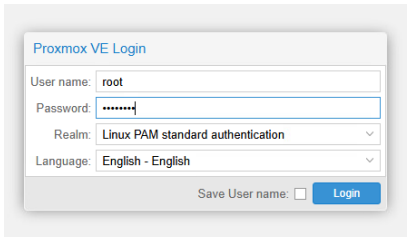


e.g.:

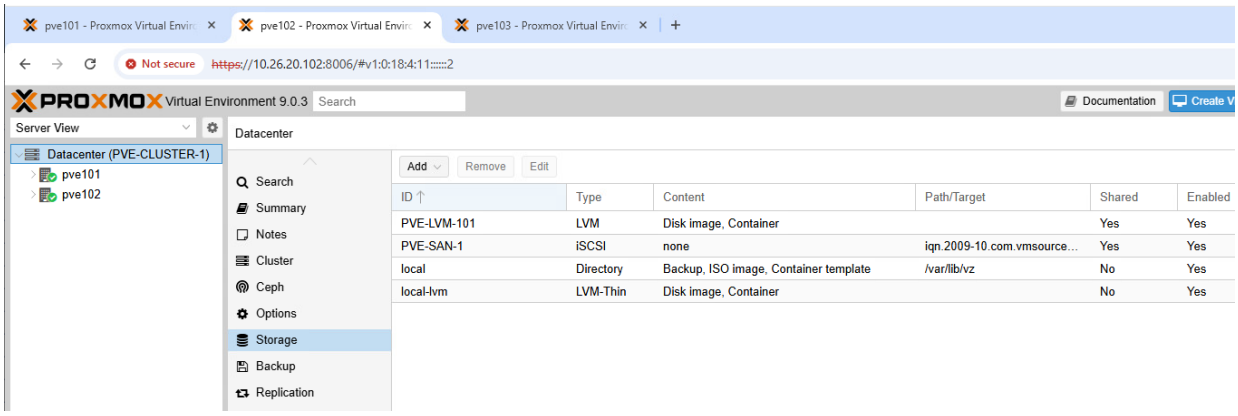


e.g.:

Step 9: Log on



e.g.:



e.g.:

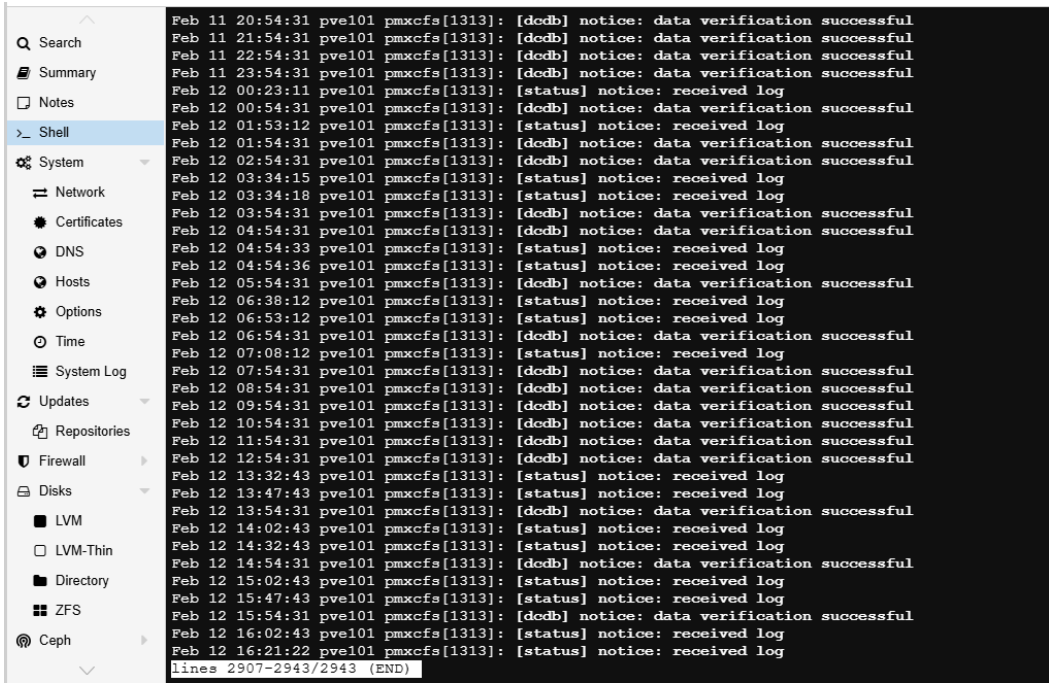
SBS LAB – (CLI) PVE Cluster diagnosis

16. Check Corosync logs for problems

Step 1: pveXYZ > Shell

Command #1 run: journalctl -b -u pve-cluster -u Corosync

e.g.:



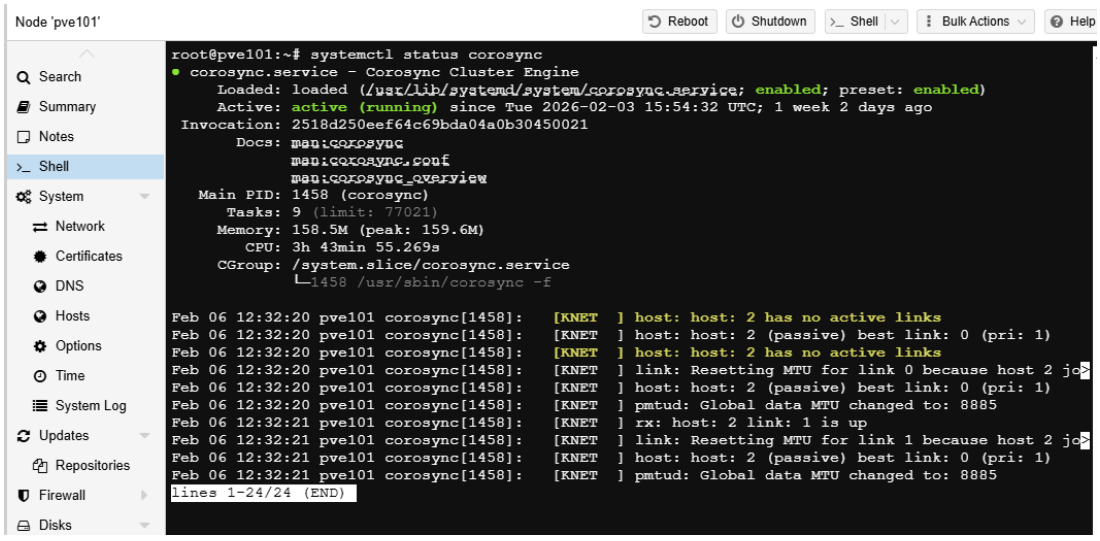
NOTE : You can enter [SHIFT]G to go to end of file

NOTE : You are looking for success/no errors

Step 2: Check Corosync Service Status

Command #1 run: systemctl status Corosync

e.g.:

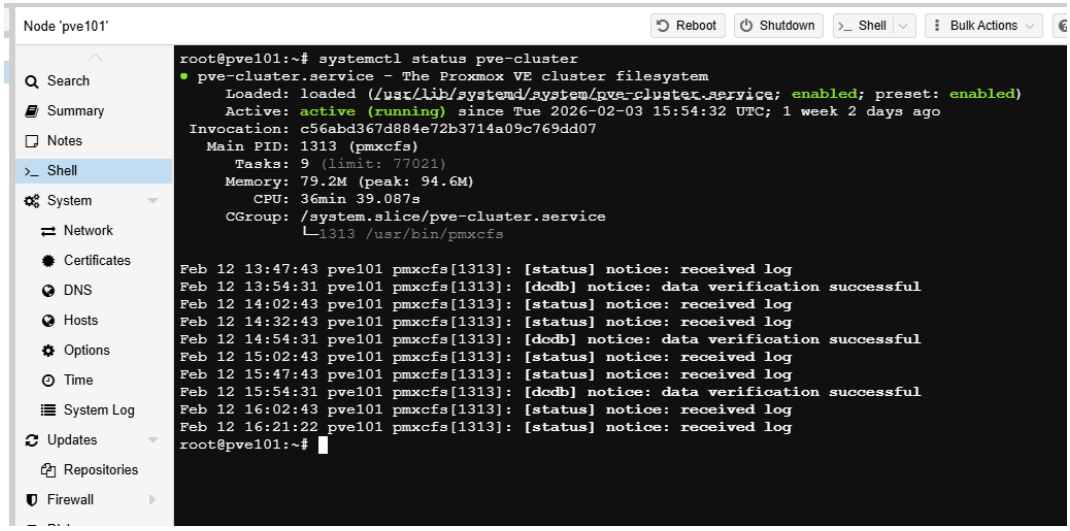


NOTE : You are looking for no active warnings/errors which are not subsequently resolved.

Step 3: Check PVE cluster services

Command #1 run: systemctl status pve-cluster

e.g.:

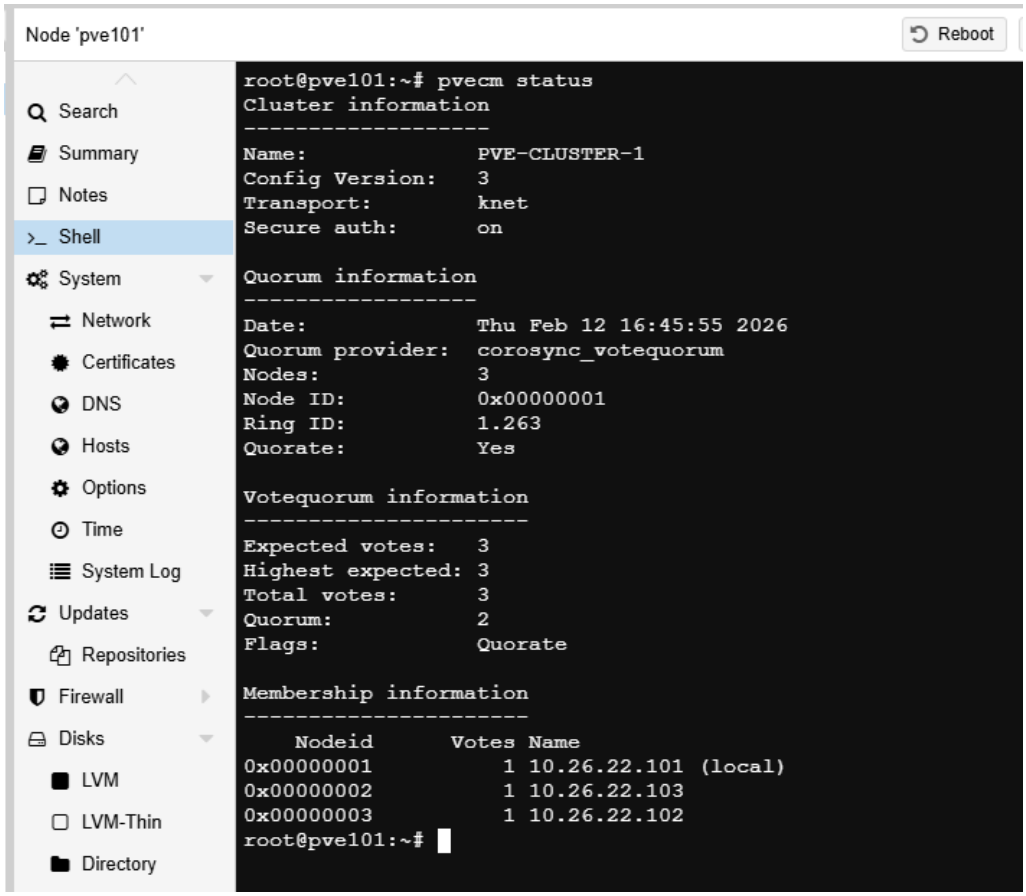


NOTE : You are looking for no active warnings/errors which are not subsequently resolved.

Step 4: Check PVE cluster status

Command #1 run: pvecm status

e.g.:



Ceph Storage for PVE

Ceph is a mature Hyperconverged storage system with no single point-of-failure. Ceph does not use a traditional filesystem but manages disks directly using a storage subsystem known as Bluestore.

Ceph clusters are created by creating OSDs on disks which present the available space within each OSD/disk added.

Ceph Monitors are then added (ideally, one monitor per node participating in the Ceph cluster) which track the liveliness of each OSD and data-placement as well.

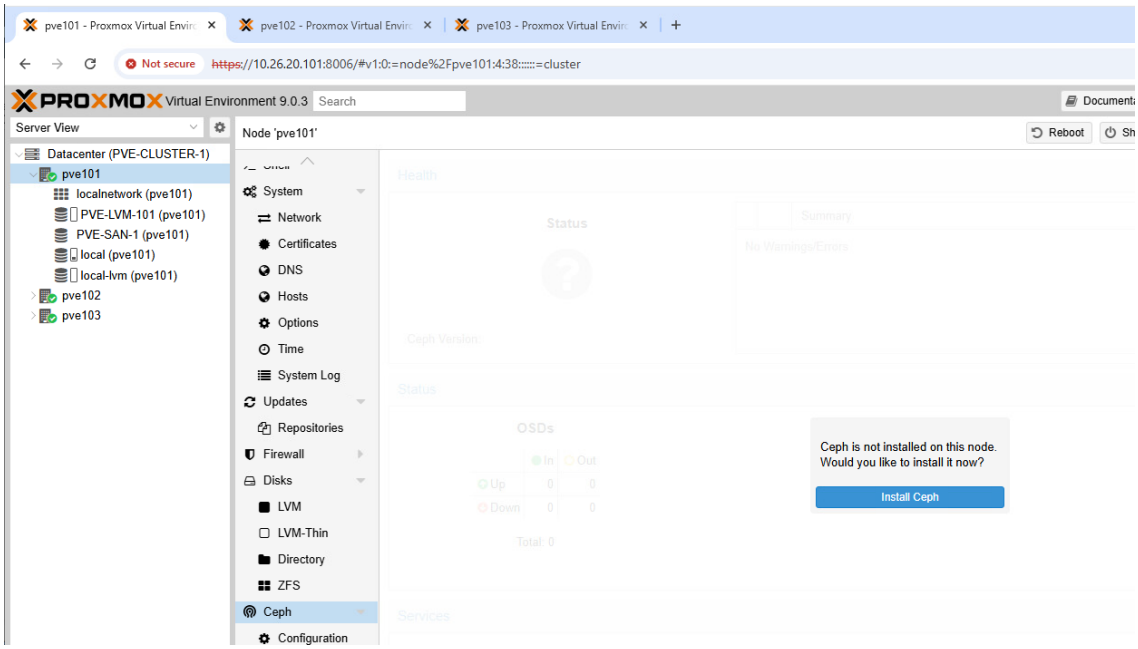
Ceph Managers are then added (ideally, one Manager per node participating in the Ceph cluster) provide the interface to the Ceph cluster.

SBS LAB – (GUI) Install Ceph

1. We need to install Ceph on our pveXYZ node

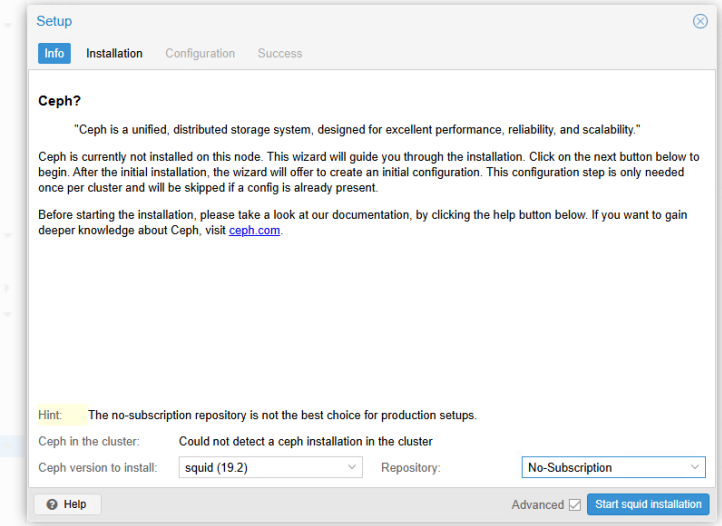
   Step 1: Install Ceph

   e.g.:



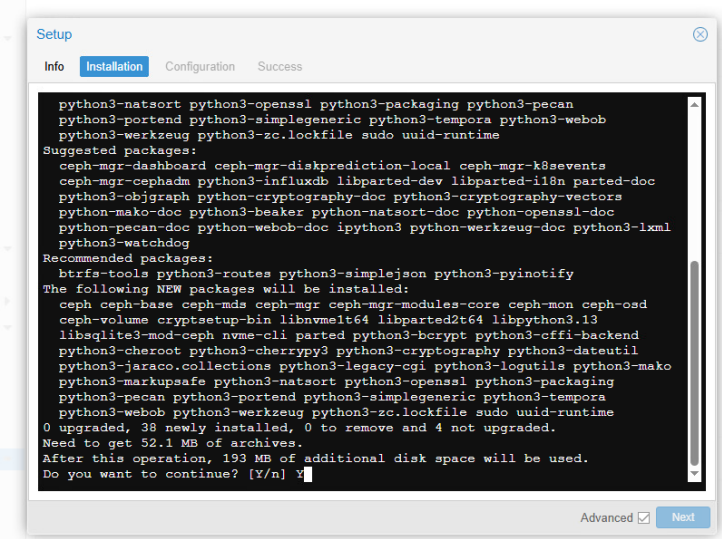
Step 2: Choose the latest Ceph version to install

Step 3: Select the No-Subscription repo



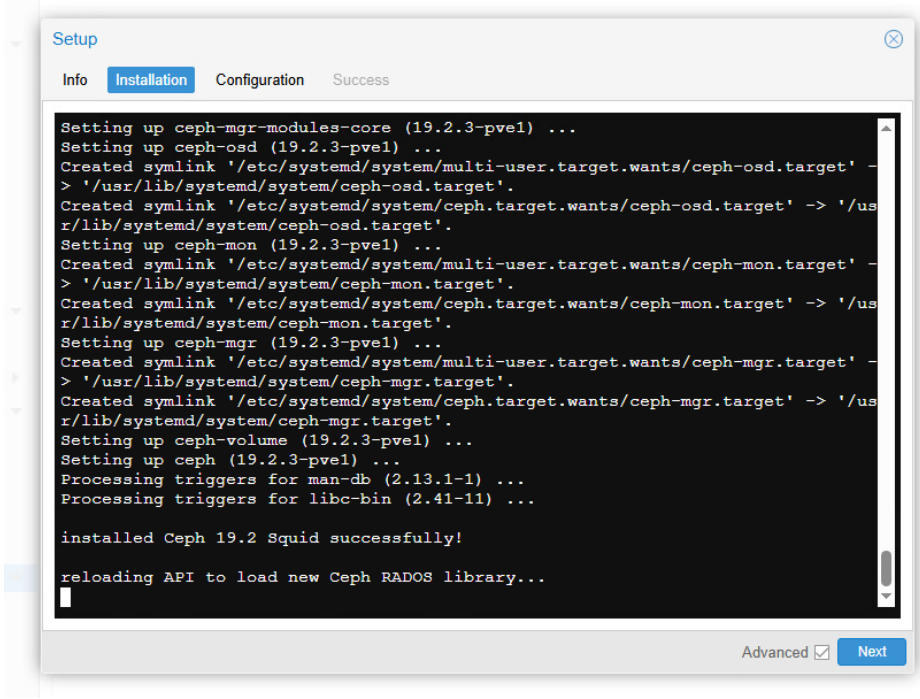
e.g.:

Step 4: [Y]



e.g.:

Step 5: Next

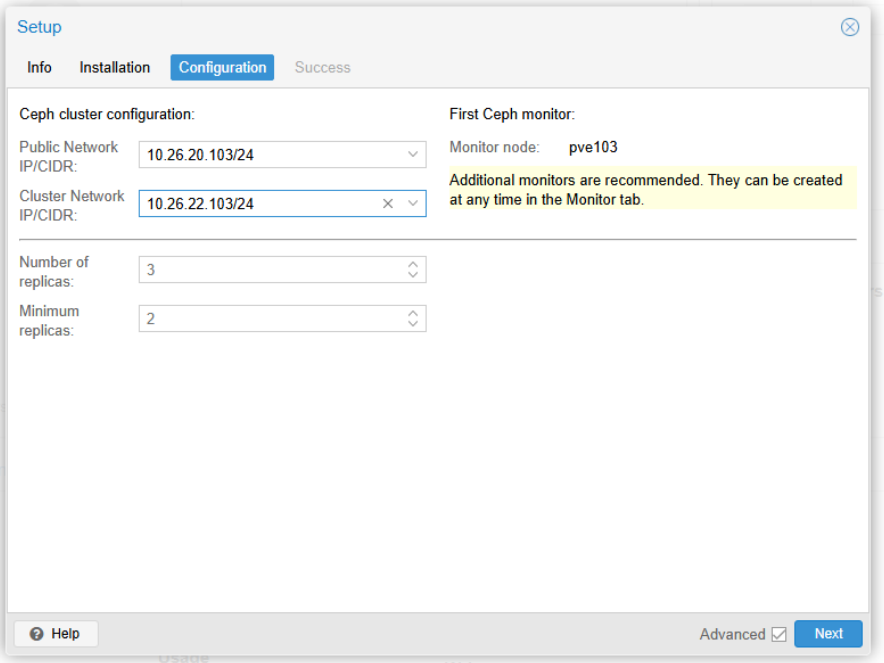


e.g.:

Step 6: Set the Public Network to our management network: 10.26.20.XYX/24

Step 7: Set the Private Network to our Ceph cluster network: 10.26.22.XYZ/24

Step 8: Next



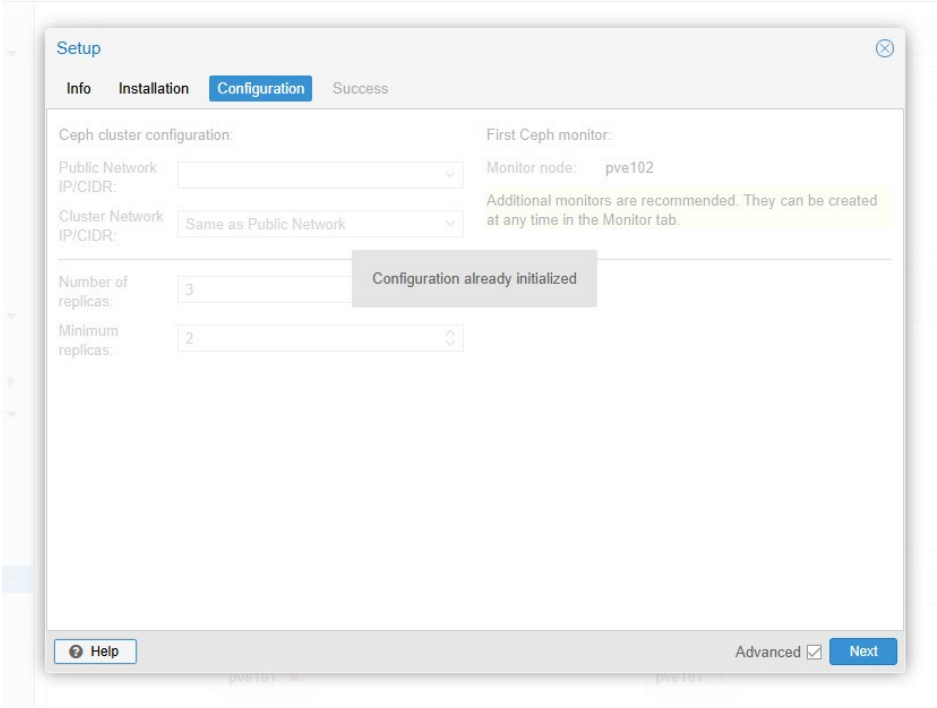
e.g.:

NOTE : Only the first Ceph cluster member gets to set the parameters

NOTE : Number of replicas represents the number of locations where data is placed (e.g. Number of replicas = 3 means that data is placed on 3 nodes). In a three node cluster, this will allow the loss of any single node.

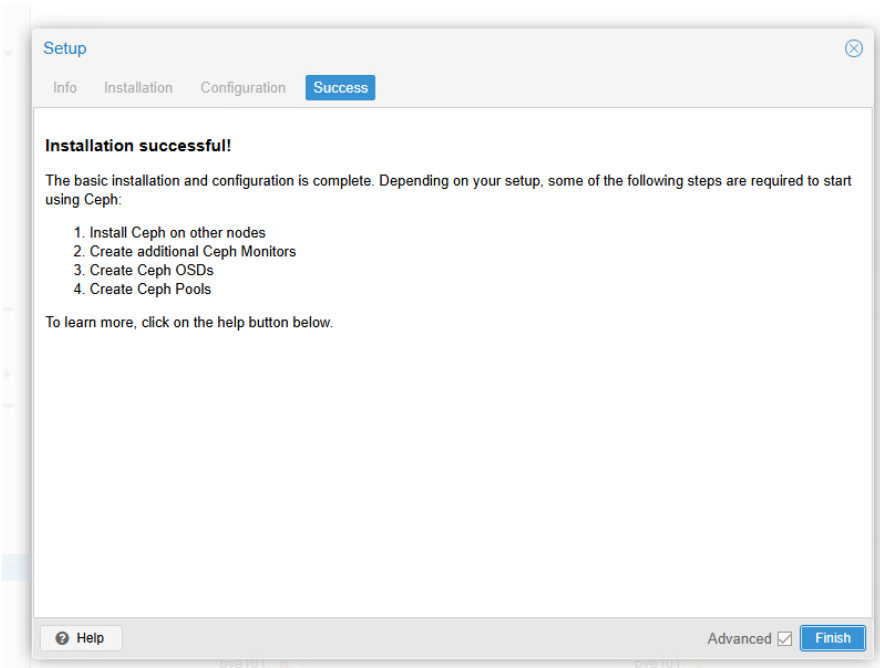
NOTE : Minimum number of replicas represents that must be available for Ceph to continue acknowledging data I/O requests. Below this size and I/O will stop to protect data integrity.

e.g.:



NOTE : All other nodes which join a Ceph cluster will see the message 'Configuration already initialized'

e.g.:

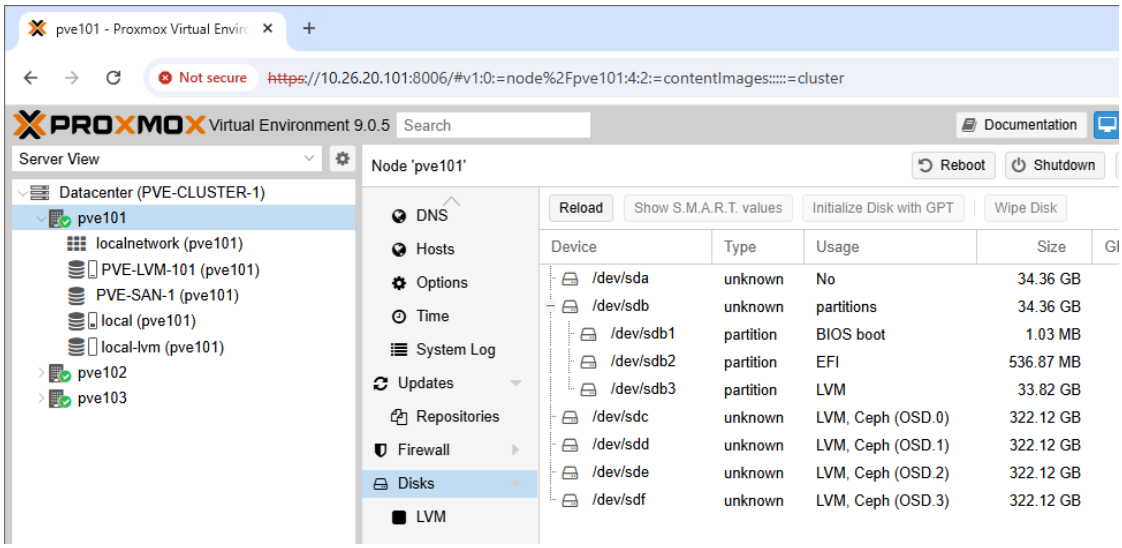


SBS LAB – (GUI) Wiping disks for Ceph

1. Before we can use physical devices for Ceph, they must be cleared of any previous data

   Step 1: pveXYZ > Disks

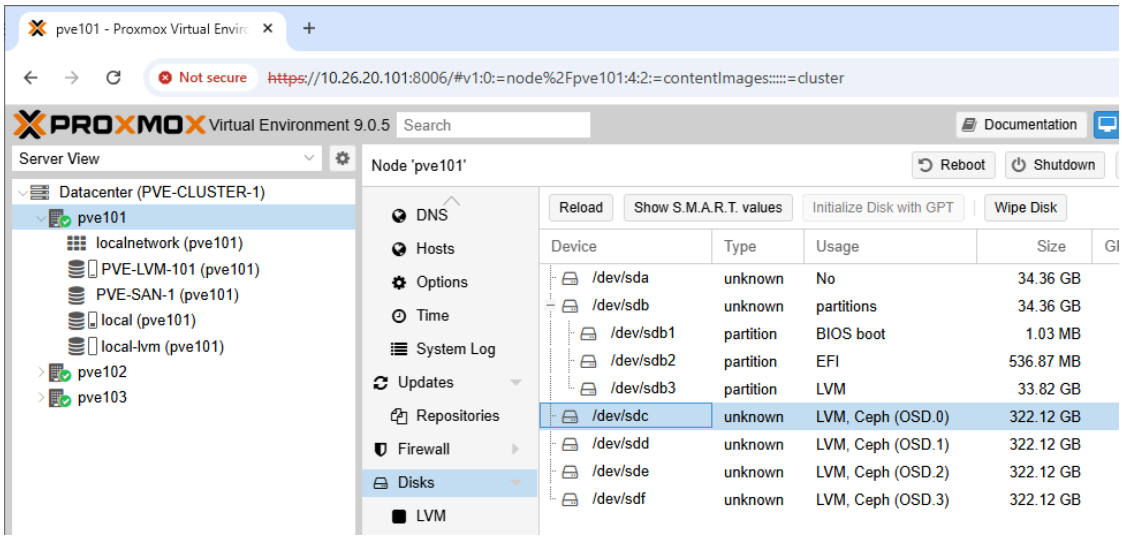
   e.g.:



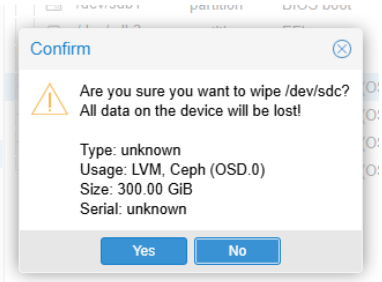
   NOTE : IF the 322GB disks have been previously used for Ceph, they will show Ceph and OSD in the usage column. **If the usage column shows No, they are ready to go, skip the rest of this SBS LAB.**

   Step 2: Select the first 322GB disk (/dev/sdc) > Wipe disk

   e.g.:



e.g.:

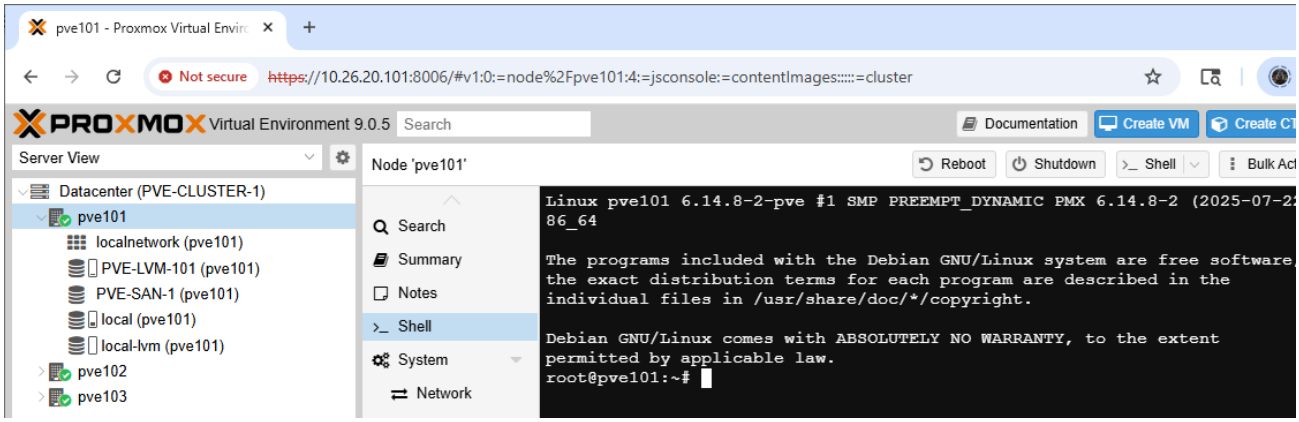


e.g.:

NOTE : Since this disk may have been previously used for Ceph, the GUI will not let us delete it.

Step 3: Open Shell

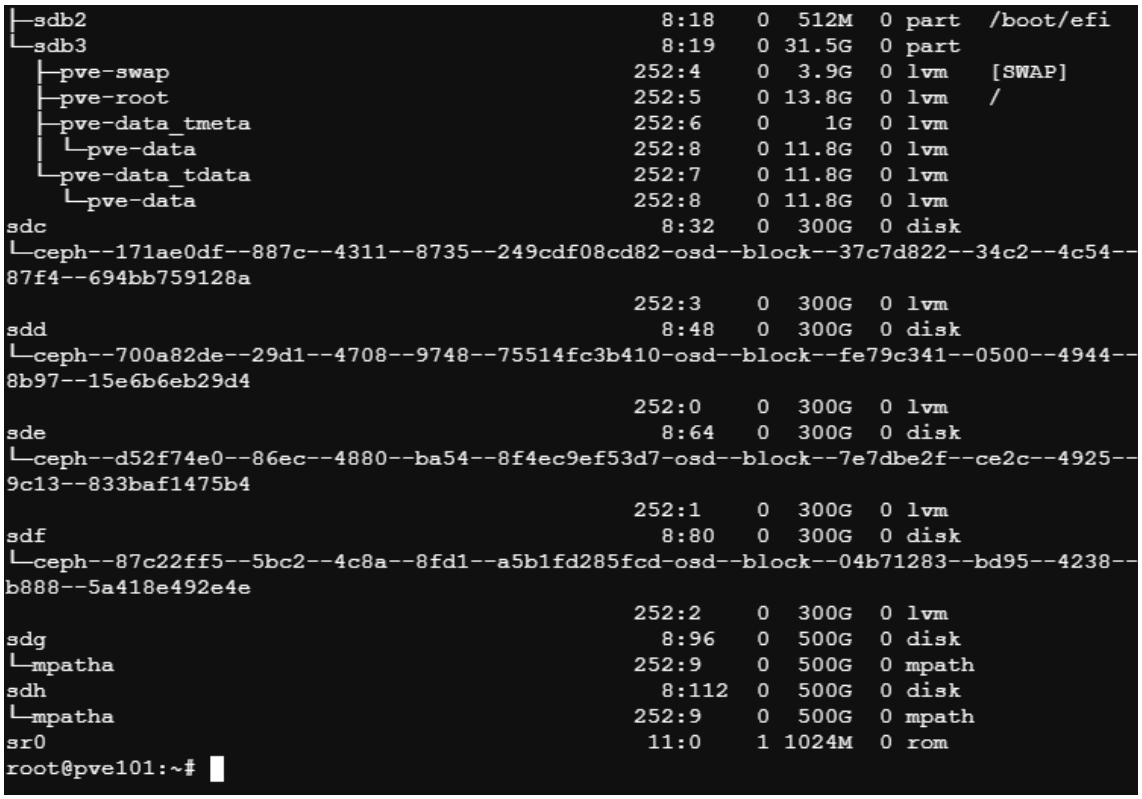
e.g.:



Step 4: List the block storage devices

Command #1 run: lsblk

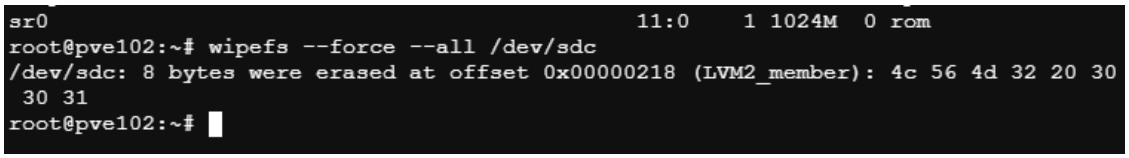
e.g.:



Step 5: Manually wipe each Ceph designated device

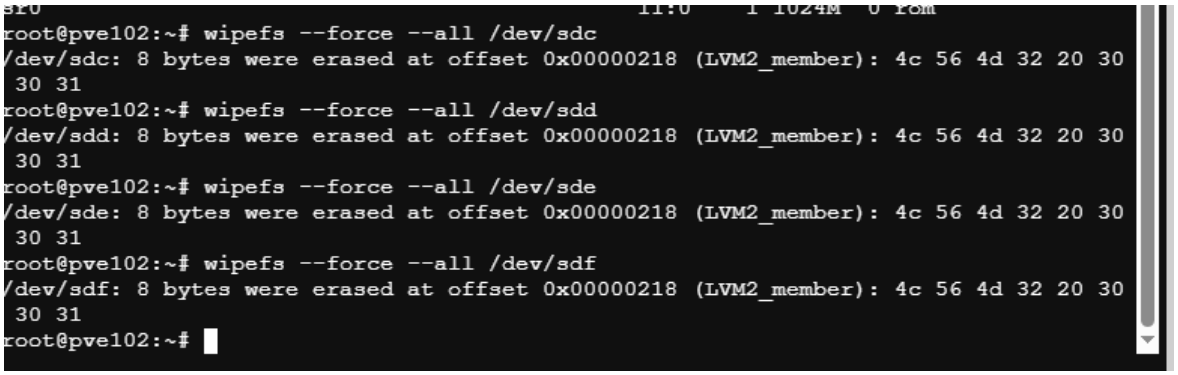
Command #1 run: wipefs --force --all /dev/sdX (where X is a Ceph disk)

e.g.:



Step 6: Repeat for all Ceph disks

e.g.:

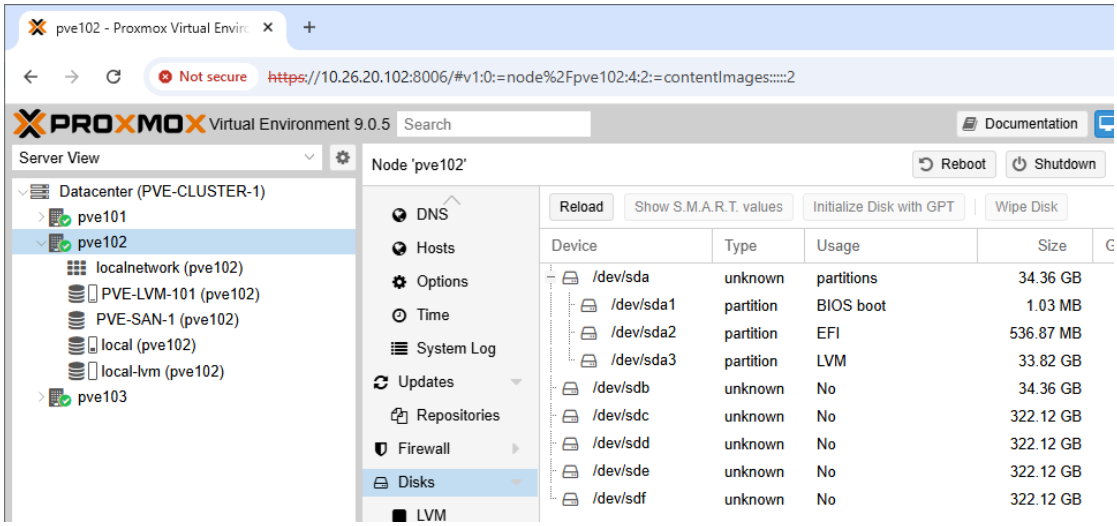


Step 7: Reboot

Command #1 run: reboot

Step 8: Check your disks again

e.g.:

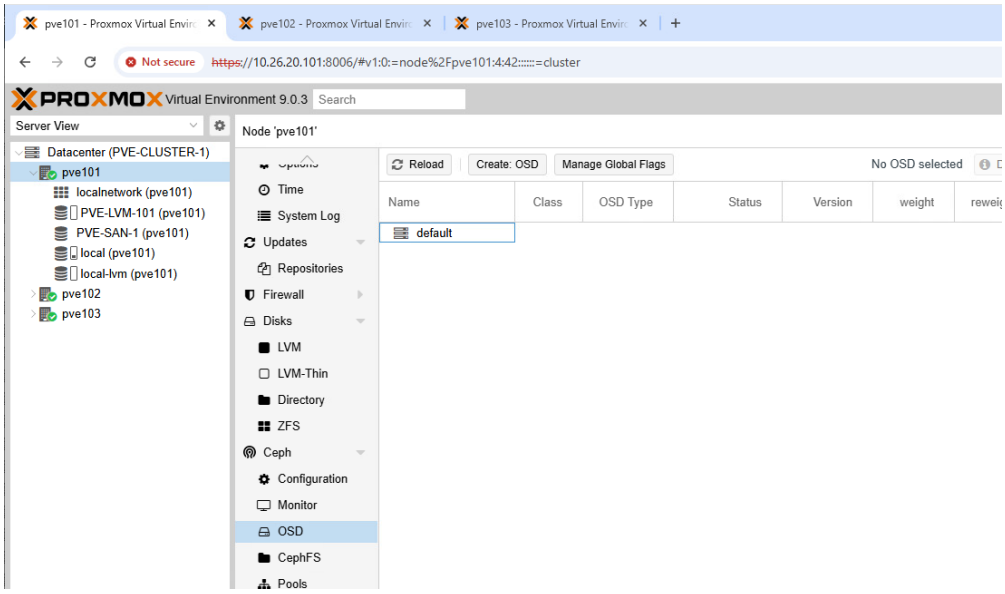


NOTE : You are now ready to create OSDs!

SBS LAB – (GUI) Create OSDs for Ceph

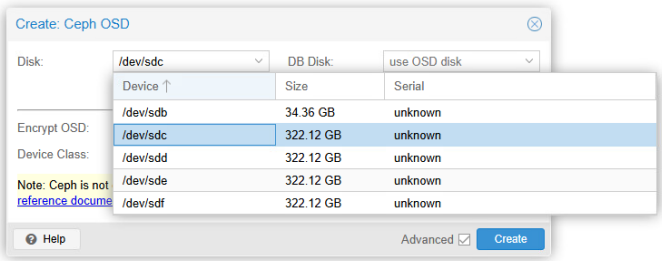
Step 1: pveXYZ > Ceph > Create OSD

e.g.:



Step 2: Select the first available Ceph disk. In our lab, all Ceph disks will be +/- 300G /dev/sdc - /dev/sdf

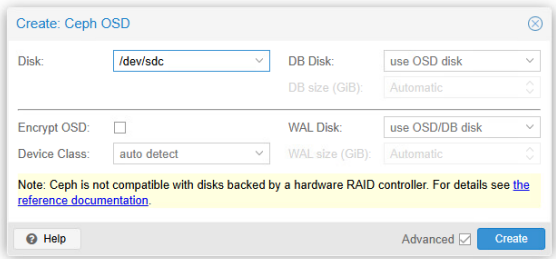
e.g.:



Step 3: Accept defaults

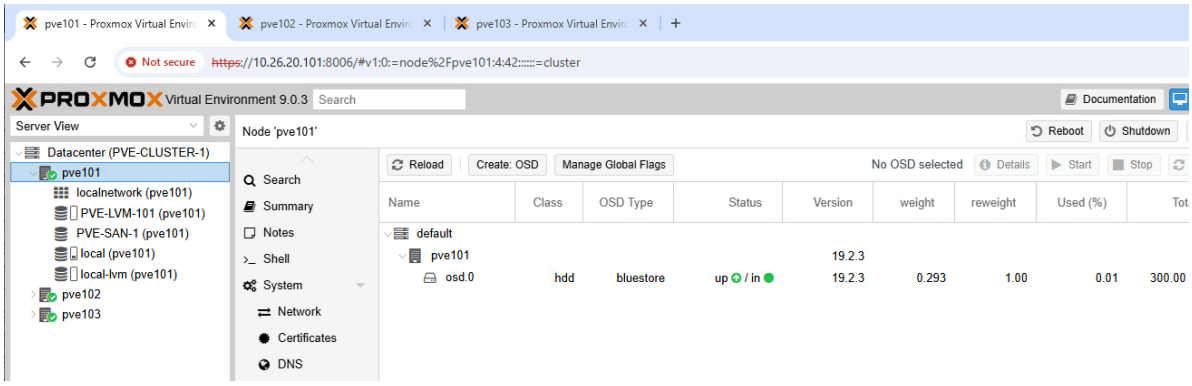
Step 4: Create

e.g.:



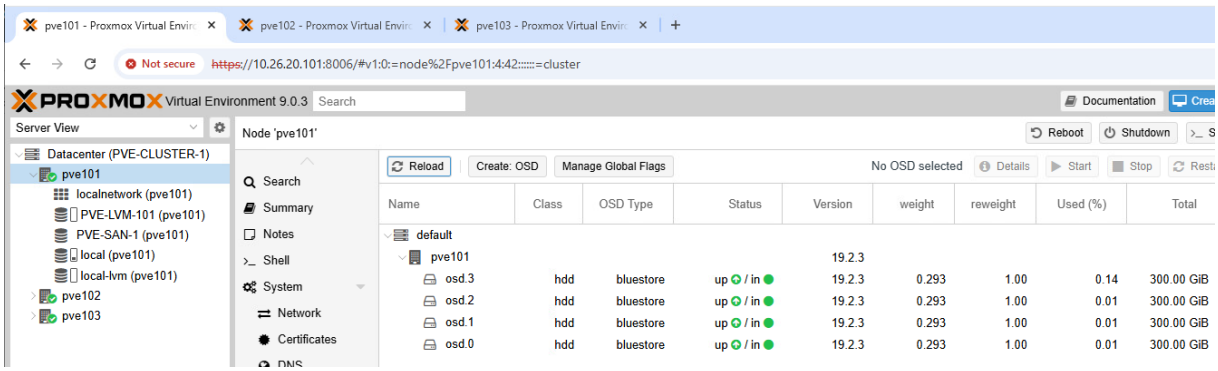
Step 5: Create an OSD for each Ceph disk on node pveXYZ you will use (we will use /dev/sdc through /dev/sdf)

e.g.:



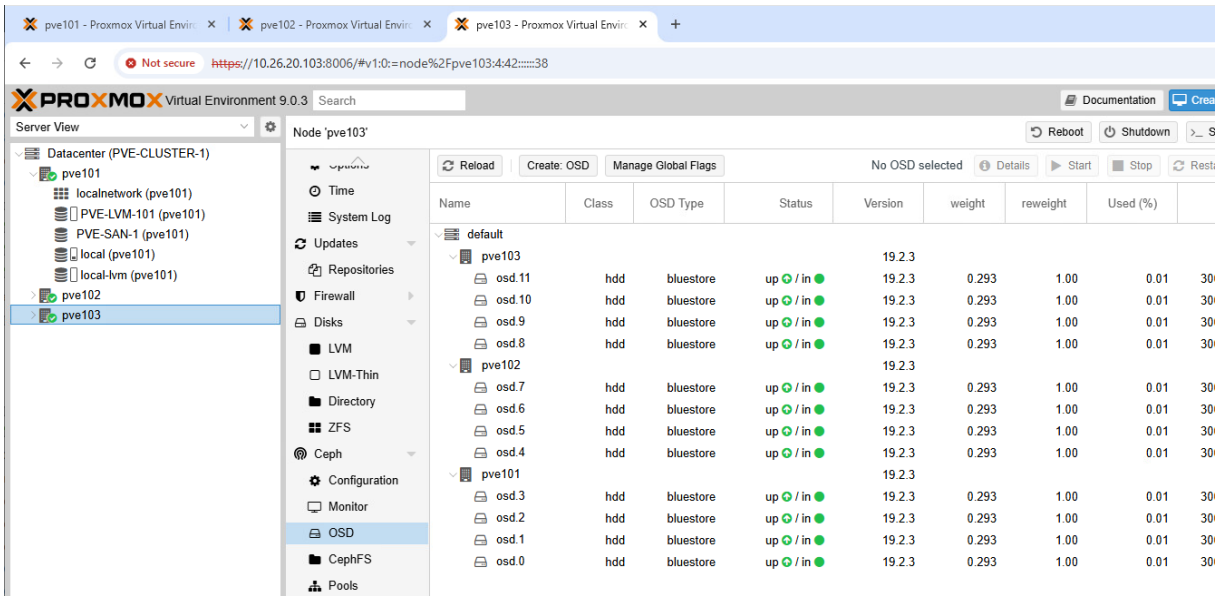
Step 6: Do this repeatedly until all Ceph disks on node pveXYZ have an OSD

Step 7:



1. A completed cluster with OSDs looks something like this (you might need to click reload to see)

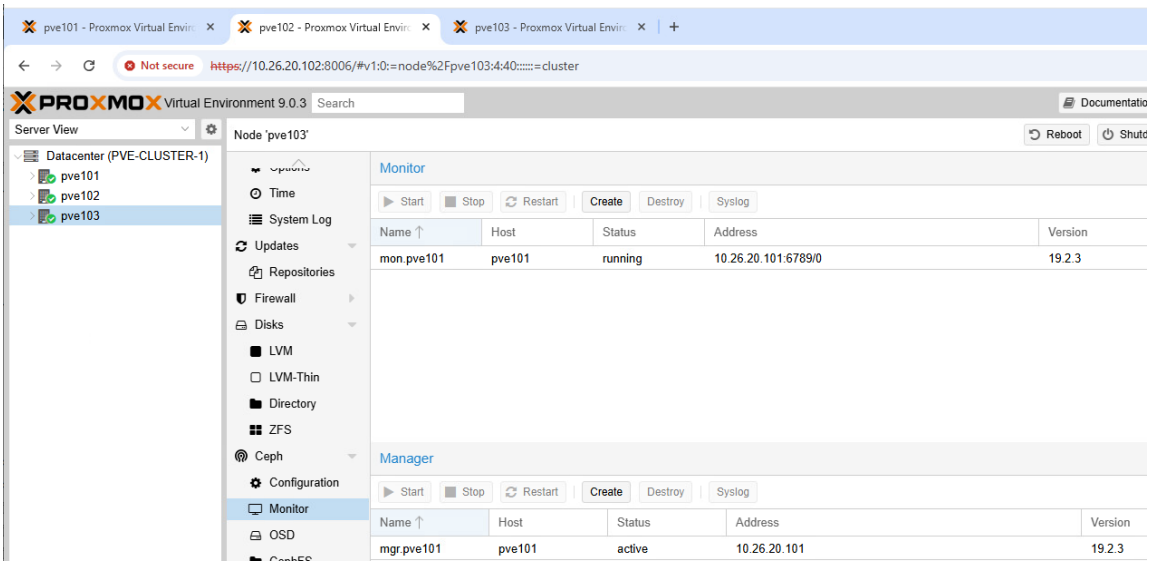
e.g.:



SBS LAB – (GUI)Create additional Ceph Monitors

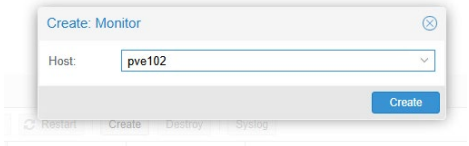
1. The first node in the cluster will automatically be the monitor node for the entire cluster. We like to create additional monitors for each node. Do this only if your node does not have a monitor listed.

   Step 1: pveXYZ > Ceph > Monitor > Create Monitor



   Step 2: Choose node pveXYZ > Create

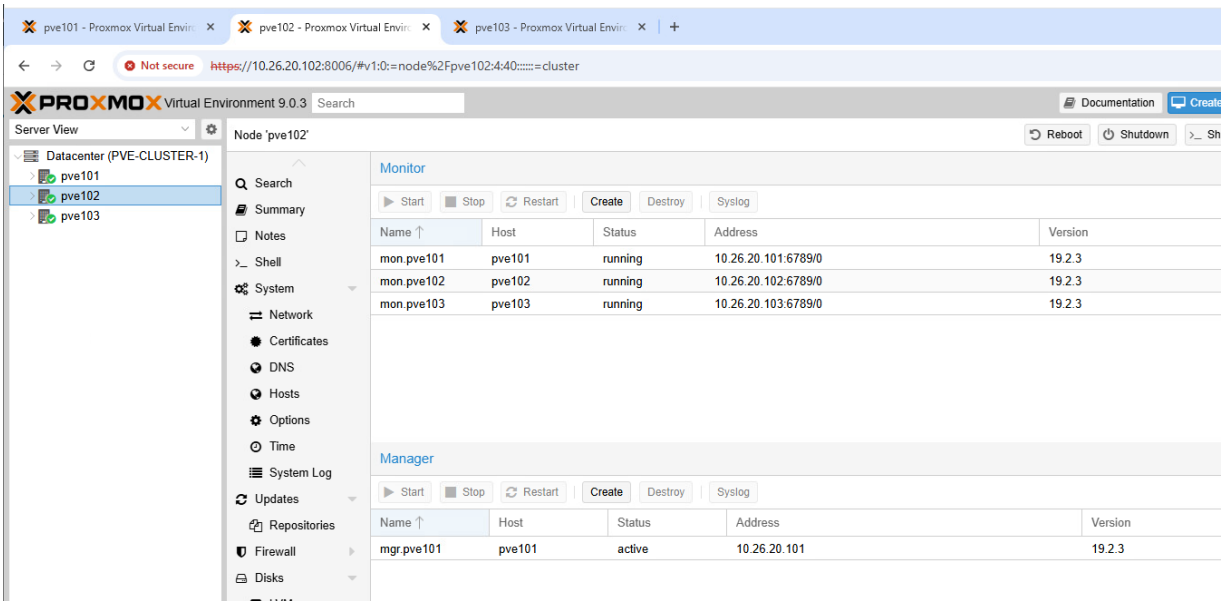
   e.g.:



SBS LAB – (GUI)Create additional Ceph Managers

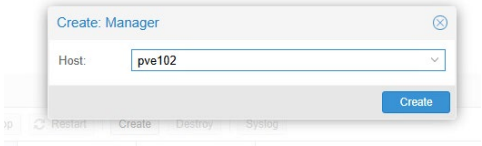
The first node in the cluster will automatically be the manager node for the entire cluster. We like to create additional managers for each node. Do this only if your node does not have a manager listed.

Step 1: pveXYZ > Ceph > Monitor > Create Manager

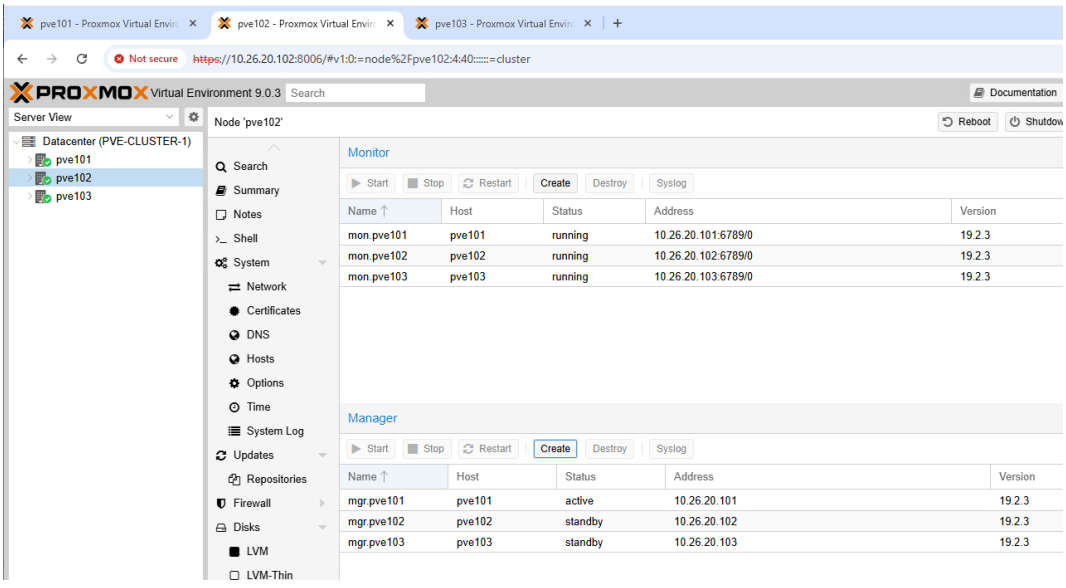


Step 2: Choose node pveXYZ

e.g.:



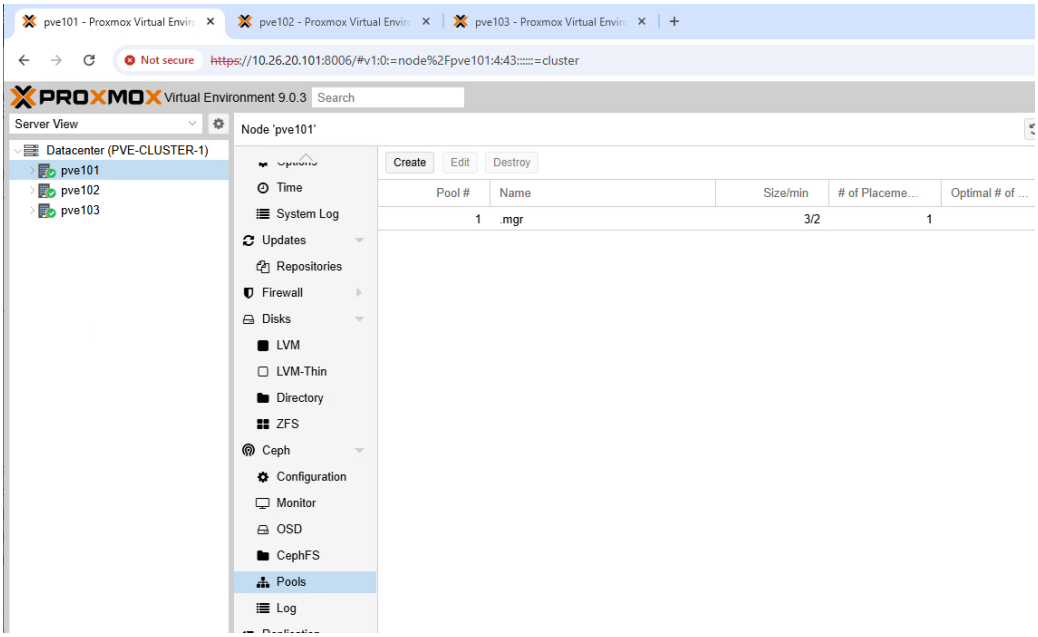
2. A completed cluster with Monitors and Managers looks something like this.

3. 

SBS LAB – (GUI) Create Ceph Pool (Only once per Cluster)

The Ceph pool is the mechanism which allows us to store VM disks on Ceph. **Only one participant (in a group environment) needs to perform the following steps**

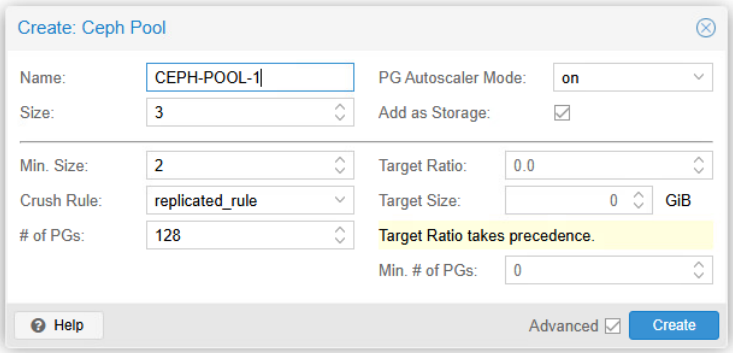
Step 1: Choose node pveXYZ > Ceph > Pools > Create



Step 2: Name the Ceph Pool: Ceph-POOL-1

Step 3: Rest of defaults are OK

Step 4: Create

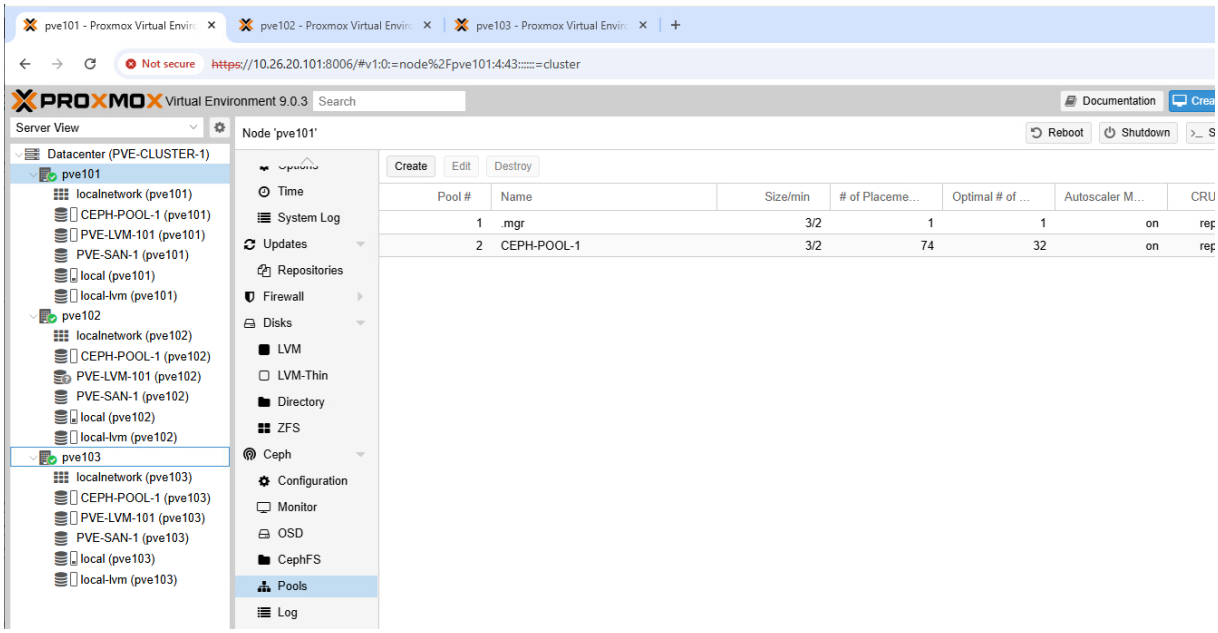


e.g.:

NOTE : PGs are Placement Groups which are used to logically distribute data across the cluster. PG Autoscaler mode will insure that there are sufficient PDs to manage the data automatically. 128 is the recommendation for small clusters (fewer than 5 nodes), with the ability to scale up if needed.

NOTE : CRUSH (Controlled Replication Under Scalable Hashing) is an algorithm used to determine data placement in Ceph. The 'replicated_rule' is specified to guarantee data availability.

e.g.:


pve101 - Proxmox Virtual Envir
pve102 - Proxmox Virtual Envir
pve103 - Proxmox Virtual Envir
Not secure
https://10.26.20.101:8006/#v1:0:=node%2Fpve101:4:43::::::=cluster
PROXMOX Virtual Environment 9.0.3
Search
Documentation
Server View
Datacenter (PVE-CLUSTER-1)
pve101
localnetwork (pve101)
CEPH-POOL-1 (pve101)
PVE-LVM-101 (pve101)
PVE-SAN-1 (pve101)
local (pve101)
local-lvm (pve101)
pve102
localnetwork (pve102)
CEPH-POOL-1 (pve102)
PVE-LVM-101 (pve102)
PVE-SAN-1 (pve102)
local (pve102)
local-lvm (pve102)
pve103
localnetwork (pve103)
CEPH-POOL-1 (pve103)
PVE-LVM-101 (pve103)
PVE-SAN-1 (pve103)
local (pve103)
local-lvm (pve103)
Node 'pve101'
Reboot
Shutdown
Time
System Log
Updates
Repositories
Firewall
Disks
LVM
LVM-Thin
Directory
ZFS
Ceph
Configuration
Monitor
OSD
CephFS
Pools
Log
Create
Edit
Destroy
Pool #
Name
Size/min
# of Placeme...
Optimal # of ...
Autoscaler M...
1 .mgr 3/2 1 1 on
2 CEPH-POOL-1 3/2 74 32 on

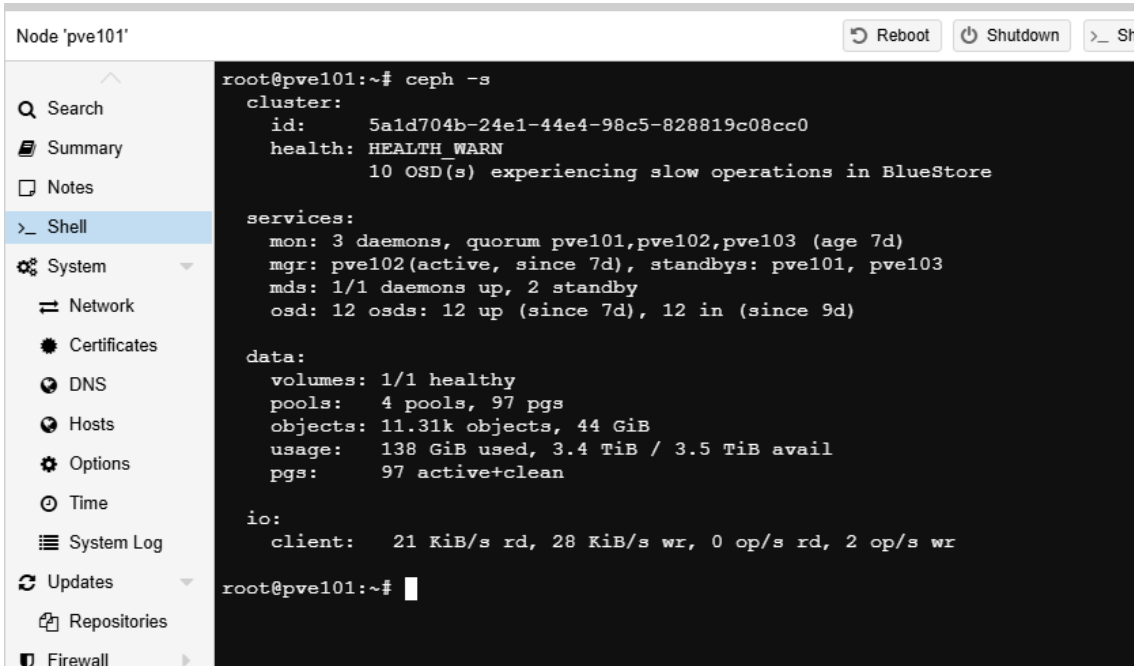
SBS LAB – (CLI) Ceph Diagnostics

From time to time it may be necessary to perform diagnostics on your Ceph storage. The following commands will help to generate data which may be used to perform diagnostics.

1. Ceph cluster status

   Command #1 run: ceph -s (Alternate command: pveceph status)

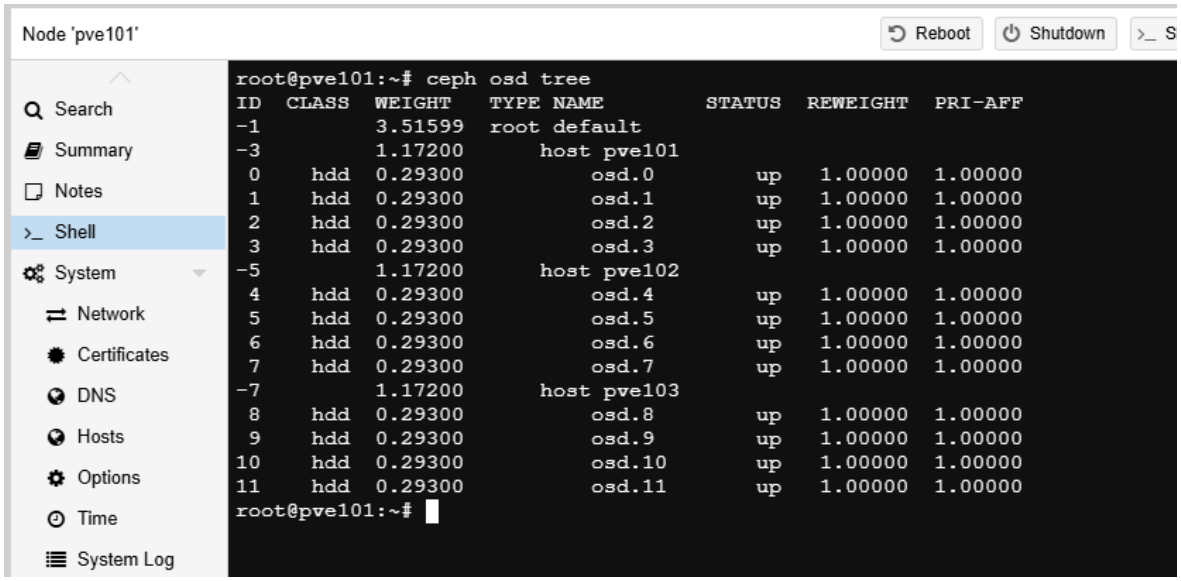
   e.g.:



2. Ceph OSD & Disk Management

   Command #1 run: ceph osd tree

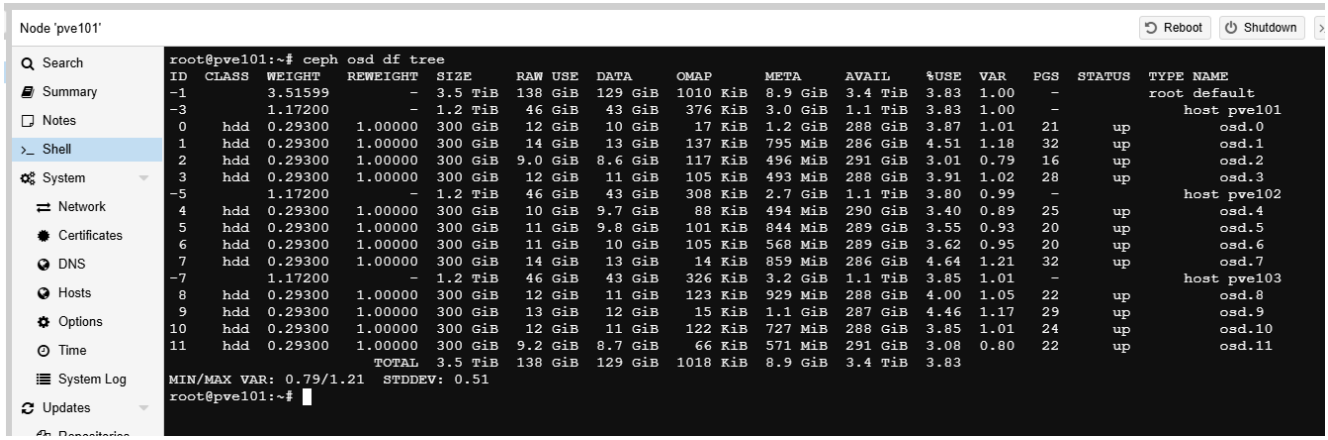
   e.g.:



   NOTE : Displays a tree view of OSDs, their status (up/down), and their WEIGHT in the CRUSH map.

Command #1 run: ceph osd df tree

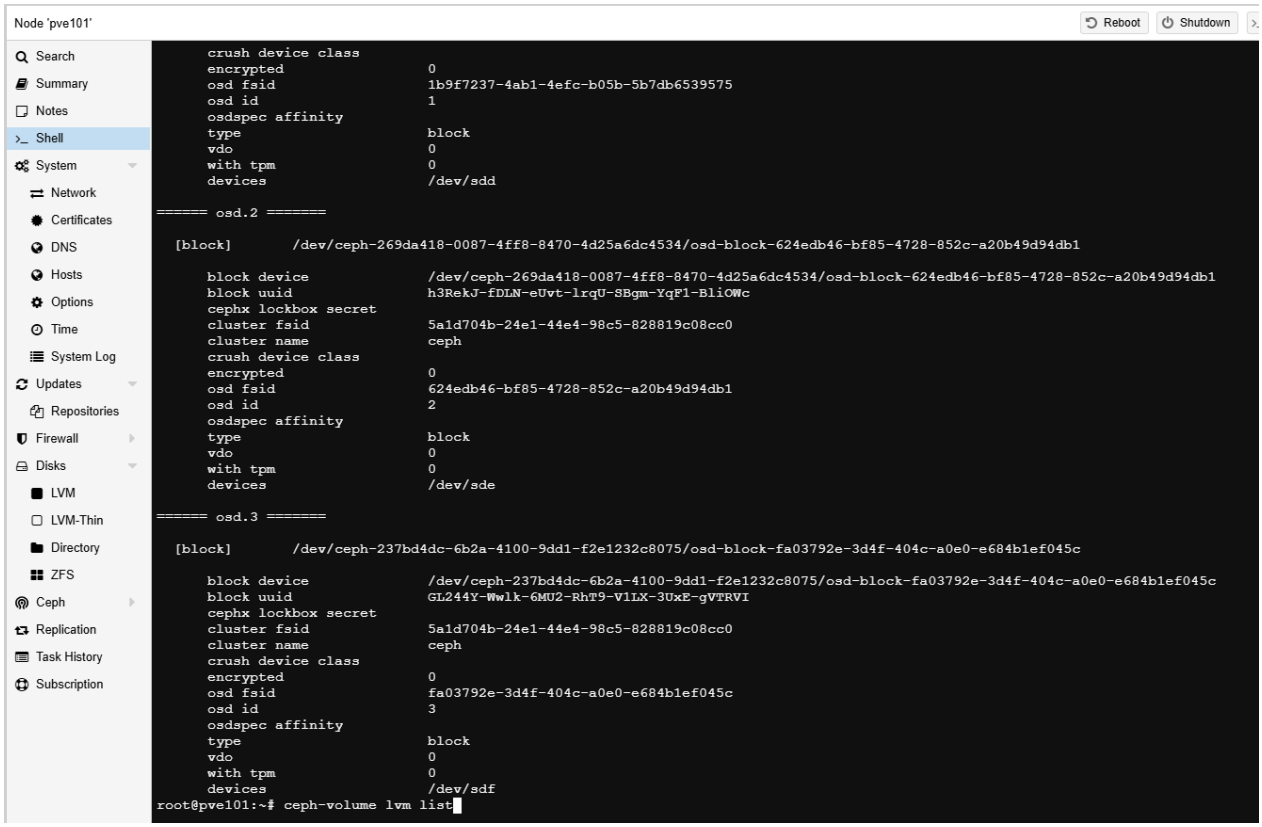
e.g.:



NOTE : Shows disk utilization per OSD

Command #1 run: ceph-volume lvm list

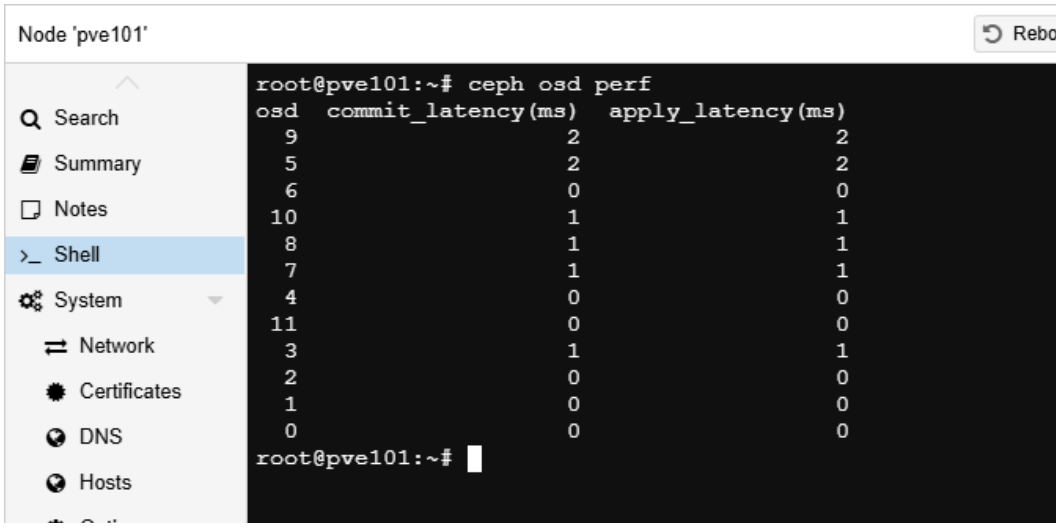
e.g.:



NOTE : Lists all OSDs and their associated LVM partitions for the node you are connected to.

Command #1 run: ceph osd perf

e.g.:

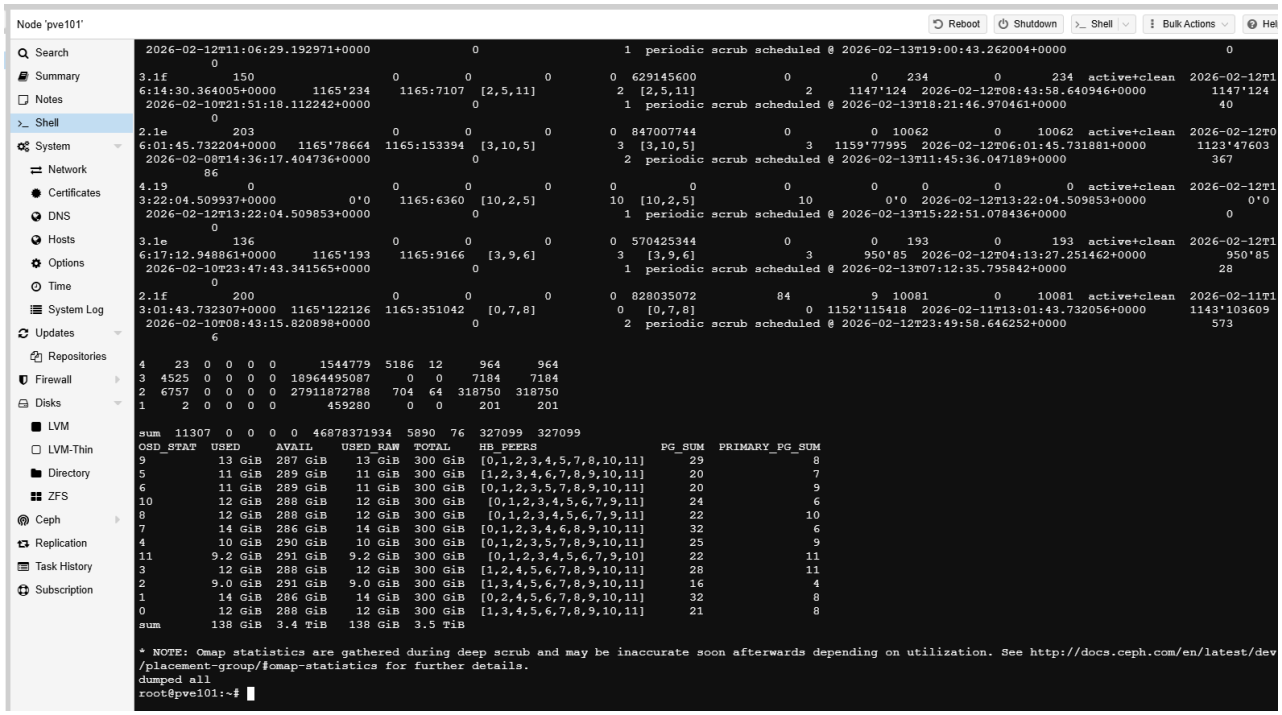


NOTE : Performs a throughput test on OSDs for the node you are connected to

3. Ceph Placement Groups (PGs) and Data Security

Command #1 run: ceph pg dump

e.g.:

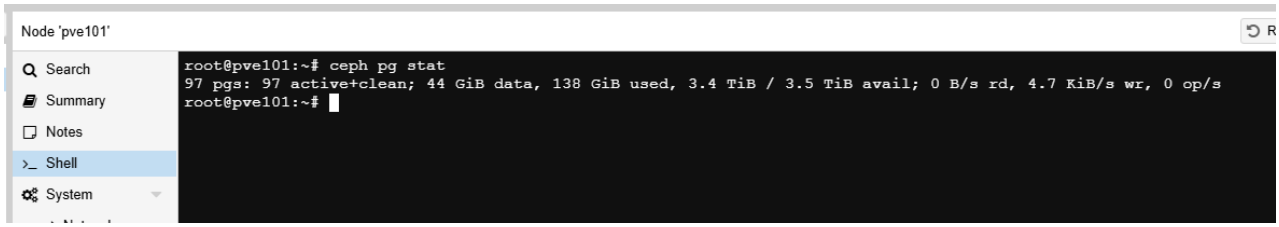


NOTE : Statistics for all placement groups

NOTE : Active+clean is good

Command #1 run: ceph pg stat

e.g.:

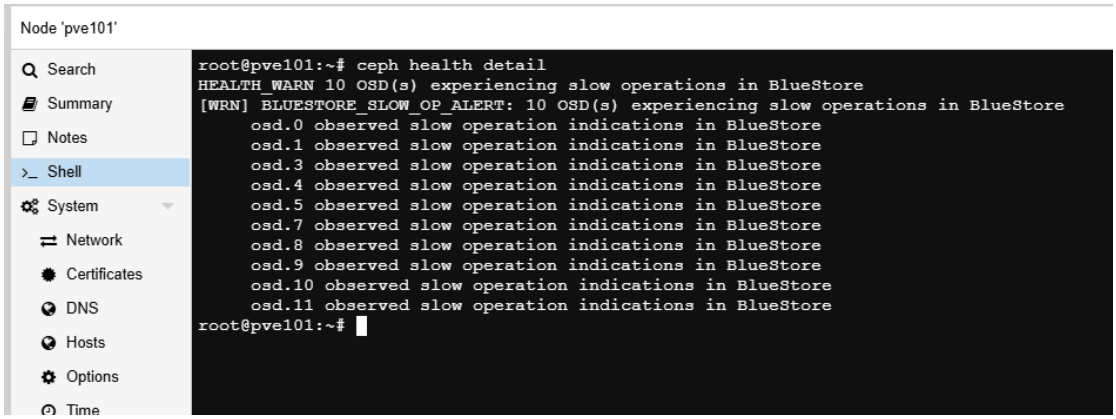


NOTE : Status of the PGs

NOTE : Active+clean is good

Command #1 run: ceph health detail

e.g.:

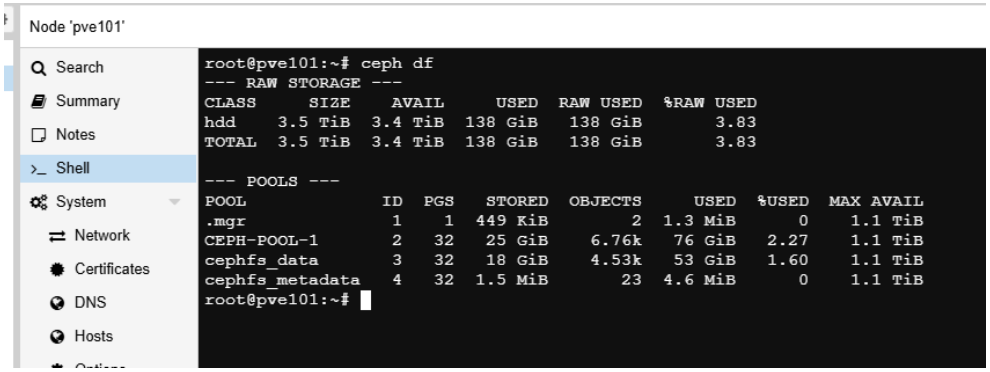


NOTE : Ceph health warnings or errors

4. Ceph Performance and Usage

Command #1 run: ceph df

e.g.:



NOTE : Usage for the entire Ceph cluster and individual pools.

Utility Storage for ISOs, Backups and more

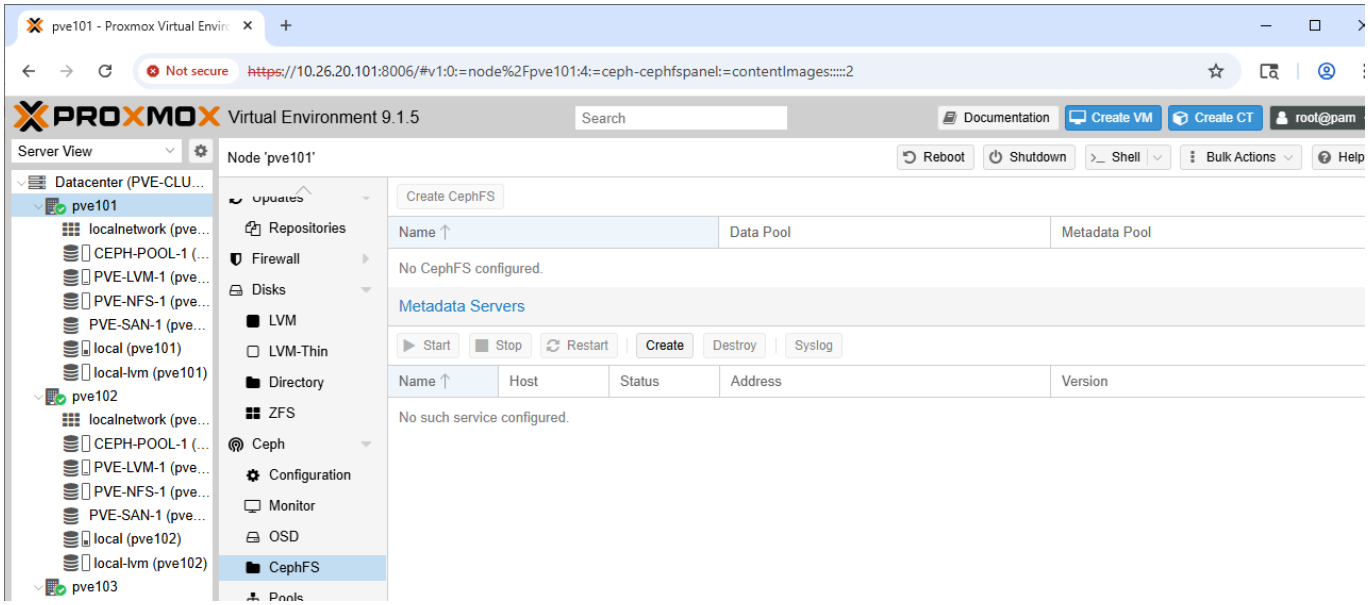
SBS LAB – Create CephFS Storage for utility

Ceph is clustered storage, but in order to use it for utility purposes (storing files), we need to create CephFS. This is a great option for ISOs, Backups and all sorts of general-purpose use. Better yet, CephFS is shared across all nodes in the cluster, so an ISO that you upload to CephFS will be available to build VMs from any node.

1. First, we create a Metadata server for CephFS

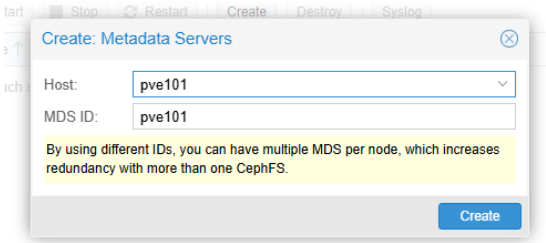
   Step 1: Go to: pveXYZ > CephFS

   e.g.:



   Step 2: Under Metadata Servers, click: Create

   e.g.:

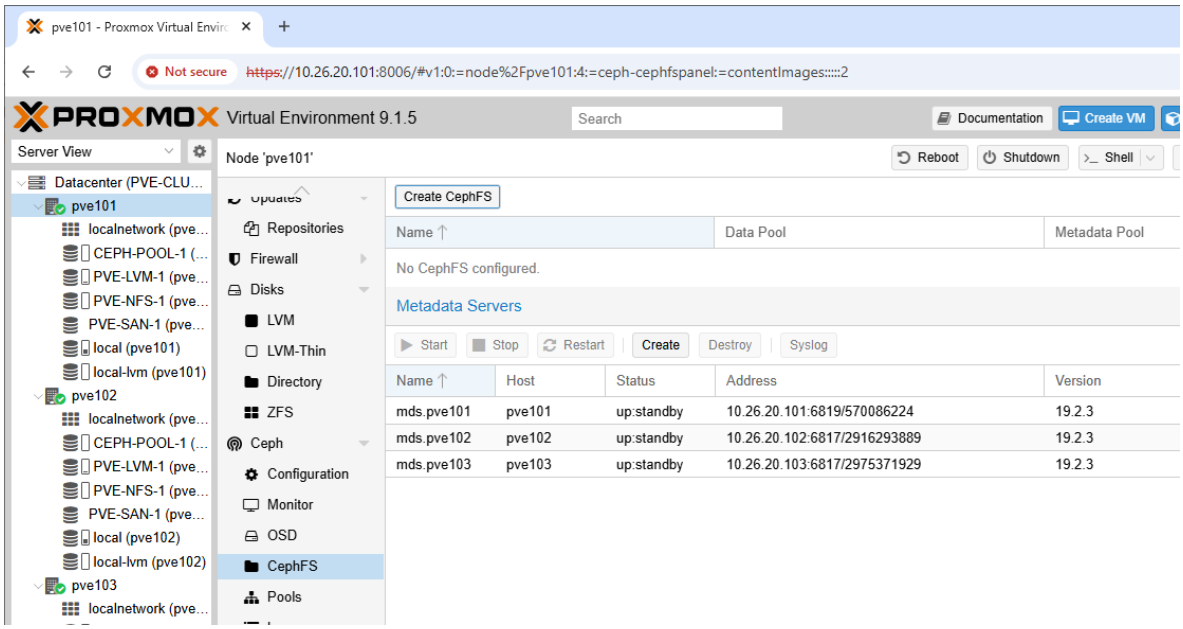


   NOTE : It is a best practice to create at least two Metadata servers, but we recommend one per node

2. Now we need to create the CephFS

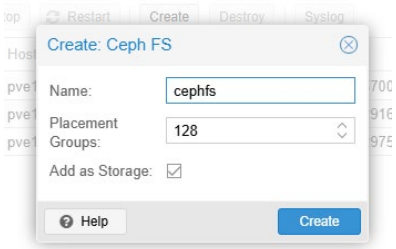
   Step 1: Click: Create CephFS

e.g.:

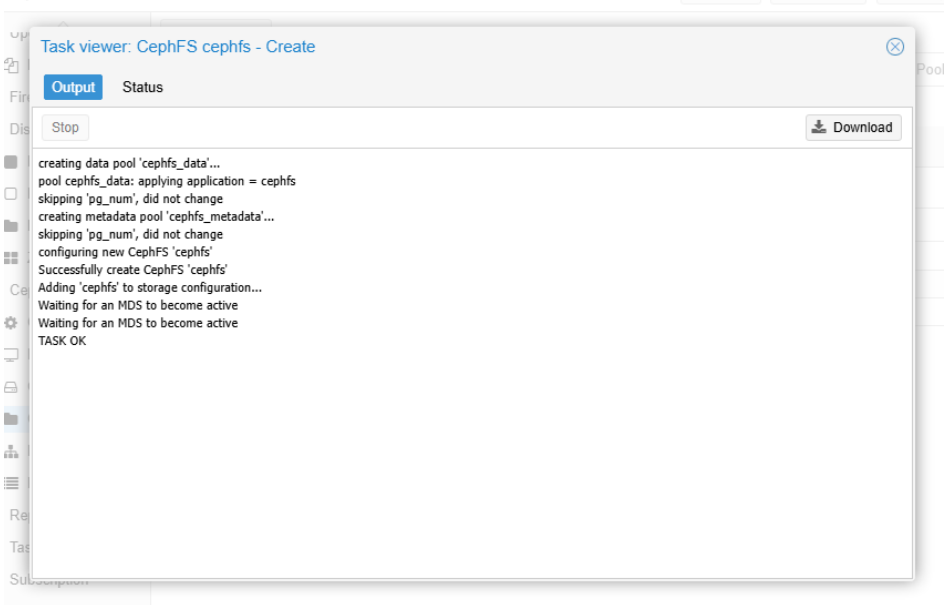


Step 2: Name it

e.g.:



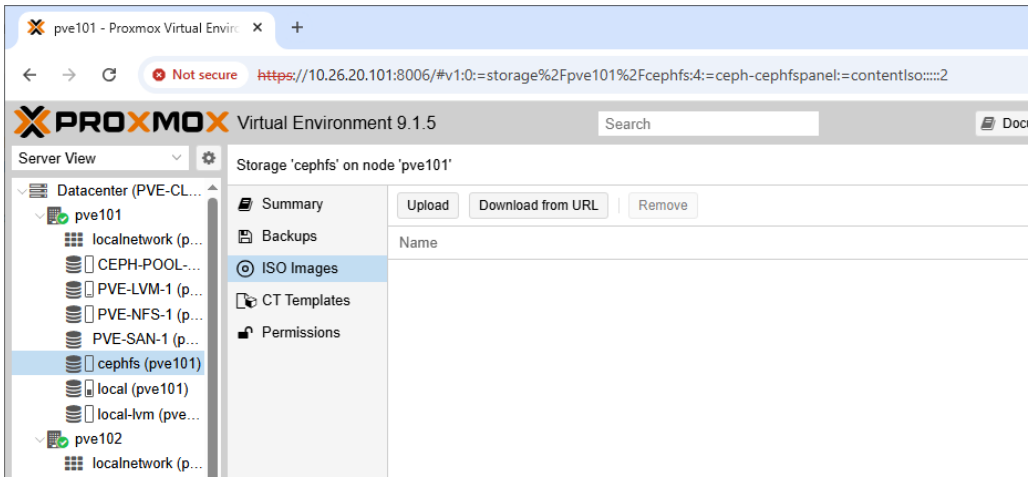
e.g.:



3. Now you can upload files to it

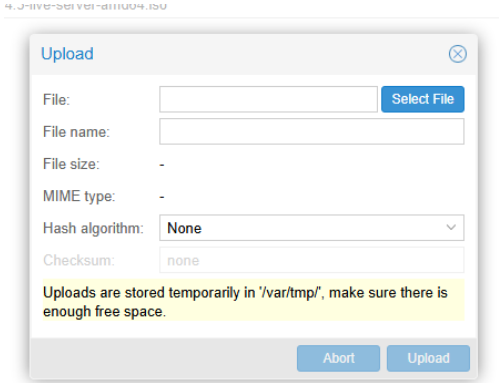
Step 1: Click Upload

e.g.:



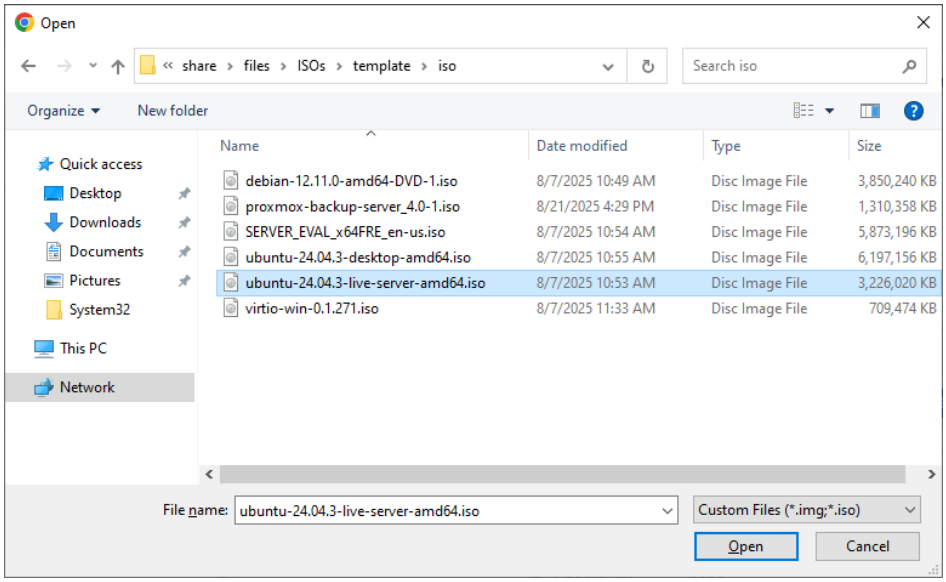
Step 2: Click select file

e.g.:



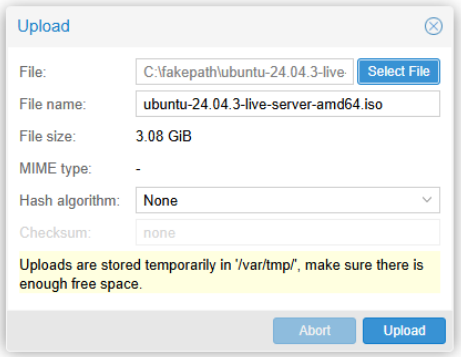
Step 3: Browse to: \\share\files\ISOs\template\iso

e.g.:

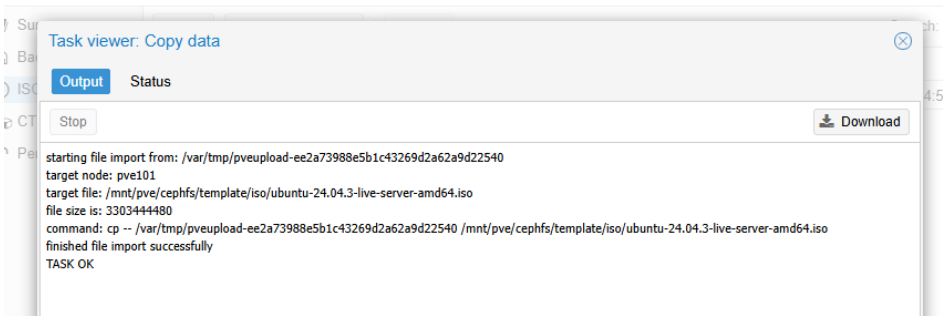


Step 4: Click Upload

e.g.:



e.g.:



4. Now the ISO is available across nodes

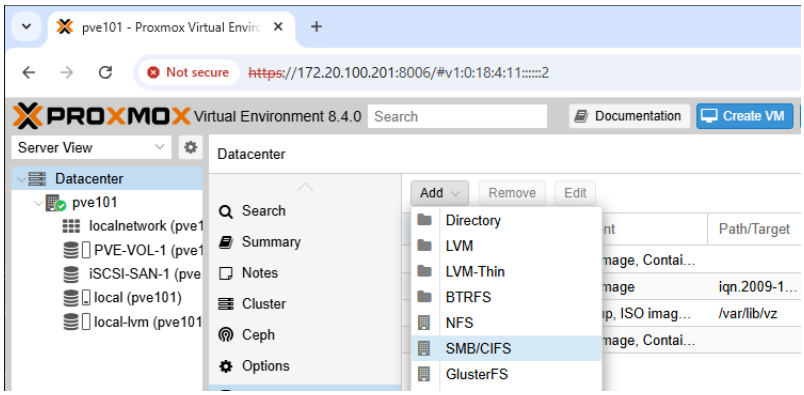
SBS LAB – (GUI) SMB/CIFS/NFS source for ISO Images

While it is certainly possible to upload your ISO images directly to a PVE node or to CephFS, many users would prefer to use a regular Windows/SMB file share for ISO images.

1. Now we are going to set up a SMB share as a source for ISO images:

   Step 1: Datacenter > Storage > Add > SMB/CIFS

   e.g.:



   Step 2: ID: ISOs

   Step 3: Server: share

   Step 4: Username: adminXYZ@lab.vmsources.com

   Step 5: Specify password

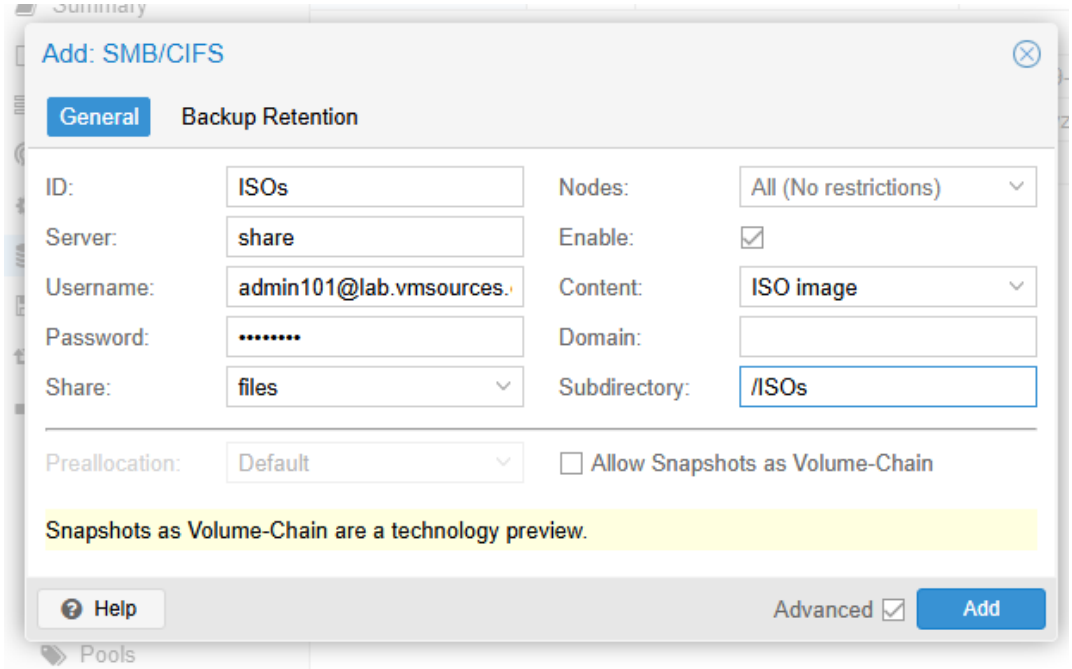
   Step 6: Share: files (choose from drop-down)

   Step 7: Content: ISO Image

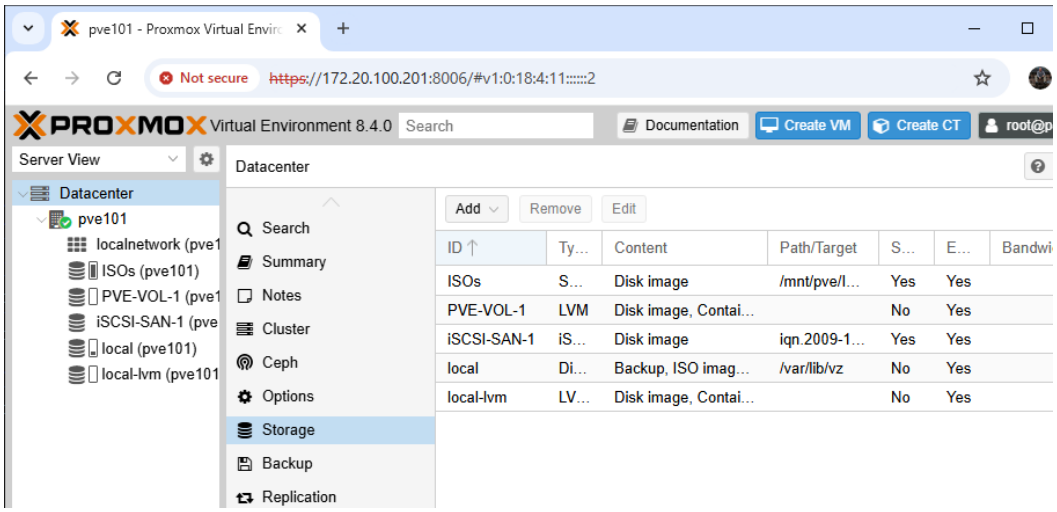
   Step 8: Subdirectory: /ISOs (omit trailing slash)

   Step 9: Add

   e.g.:

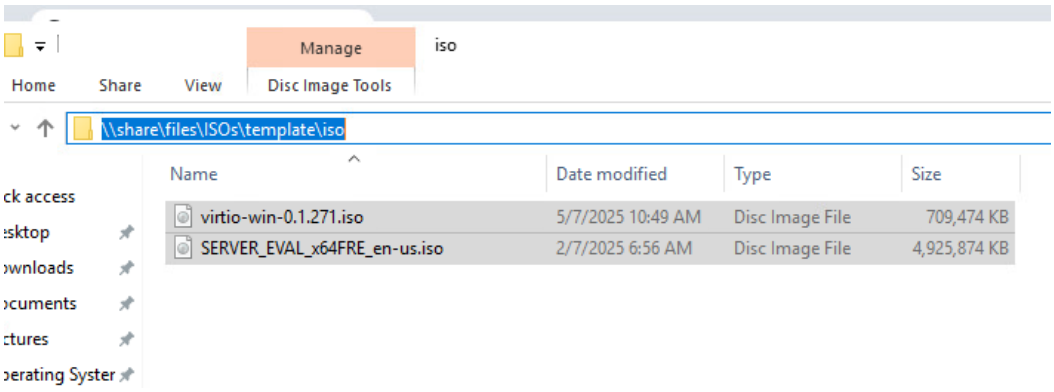


e.g.:



NOTE : PVE will create folders on your SMB/CIFS Share, we have pre-populated the folder with the ISOs used here. The full SMB folder path is: <SMBPATH>\template\iso

e.g.:

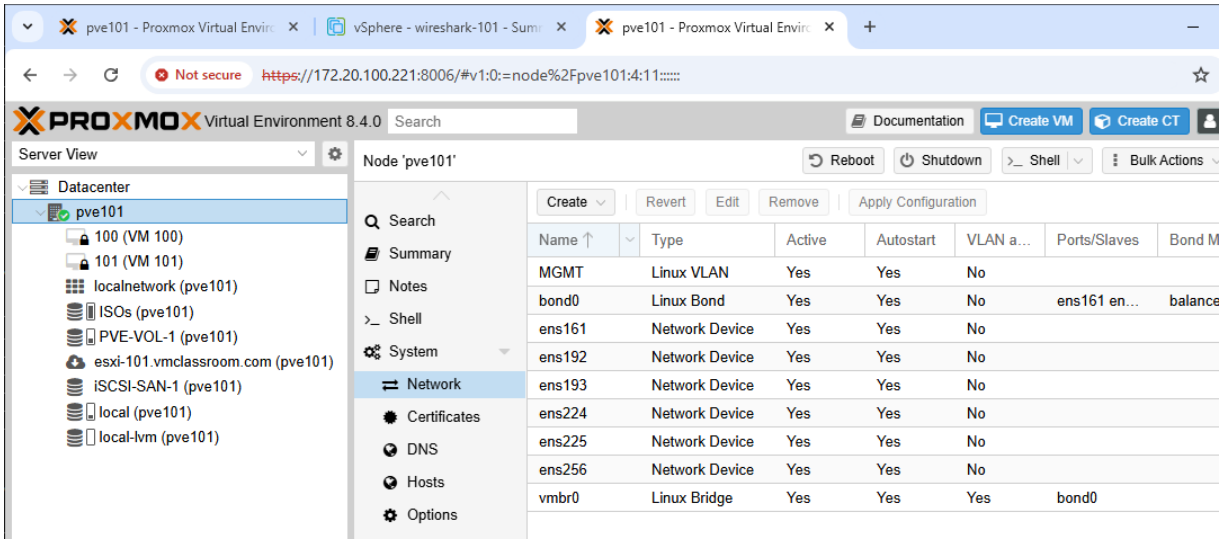


Building and Managing VMs

SBS LAB –(GUI) Building a Windows VM

Step 1: Click on > Create VM

e.g.:

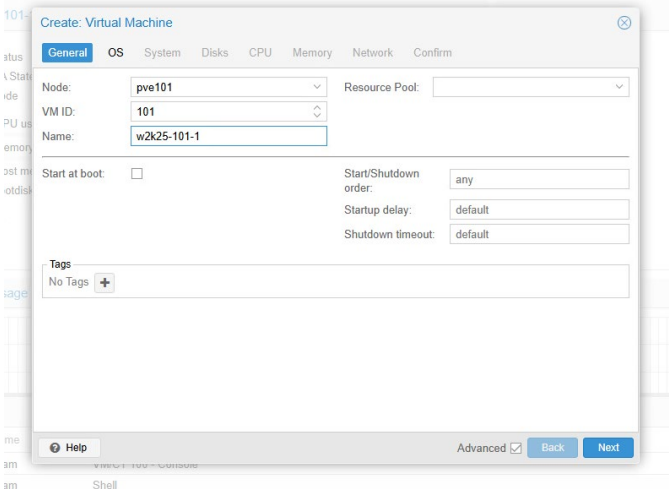


Step 2: Choose node: pveXYZ

Step 3: Set the VM friendly name: w2k25-XYZ-1

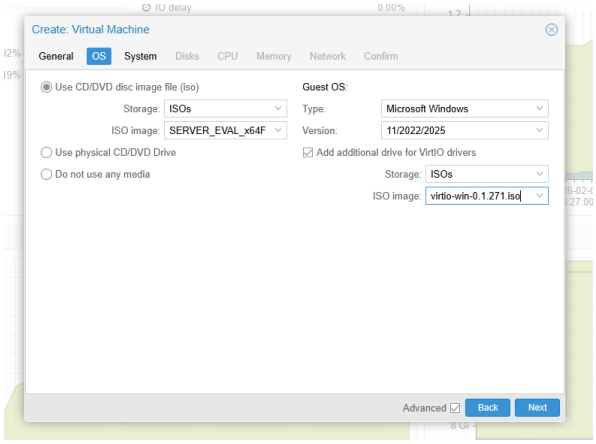
Step 4: Next

e.g.:



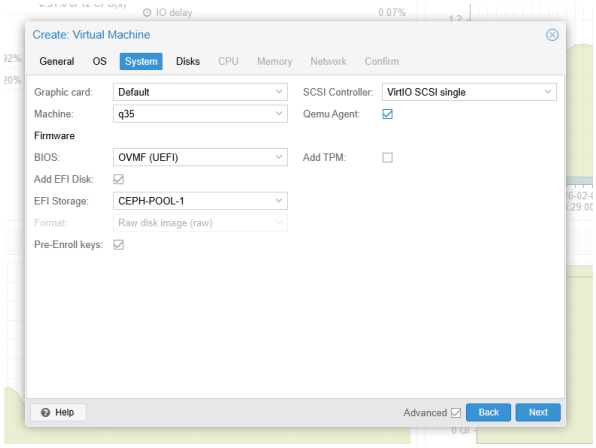
Step 5: Storage: ISOs
Step 6: ISO Image: SERVER_EVAL...
Step 7: Type: Microsoft Windows
Step 8: Version: latest
Step 9: All additional disk for VirtIO drivers: checked
Step 10: Storage: ISOs
Step 11: ISO Image: virtio-win....
Step 12: Next

e.g.:



Step 13: Use defaults in most places *EXCEPT*
Step 14: Choose EFI Storage (presumably iSCSI SAN volume)
Step 15: Qemu Agent: checked
Step 16: Add TPM: unchecked
Step 17: Next

e.g.:



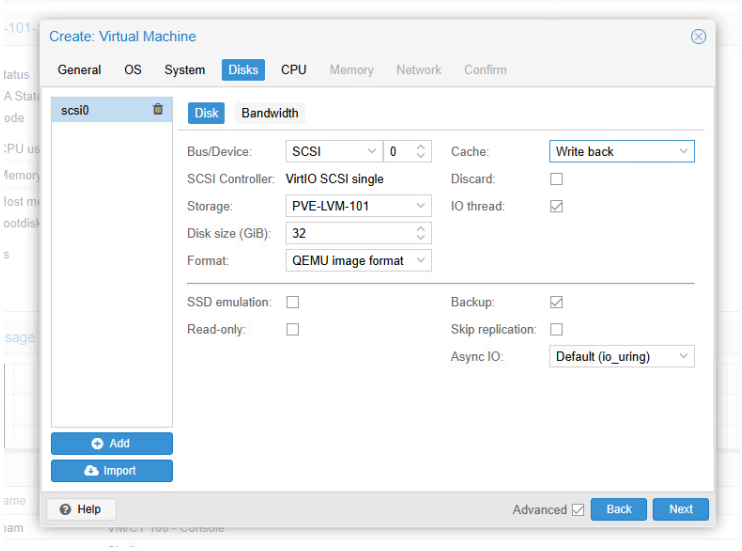
Step 18: Use defaults in most places *EXCEPT*

Step 19: Storage: PVE-LVM-XYZ

Step 20: Cache: Write-back (fast but safe)

Step 21: Next

e.g.:



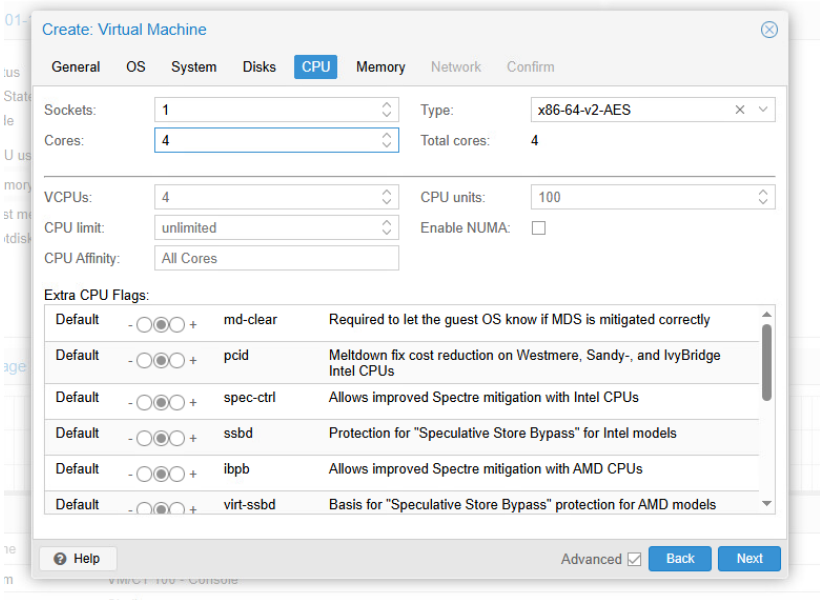
Step 22: Use defaults in most places *EXCEPT*

Step 23: Sockets: 1

Step 24: Cores: 4

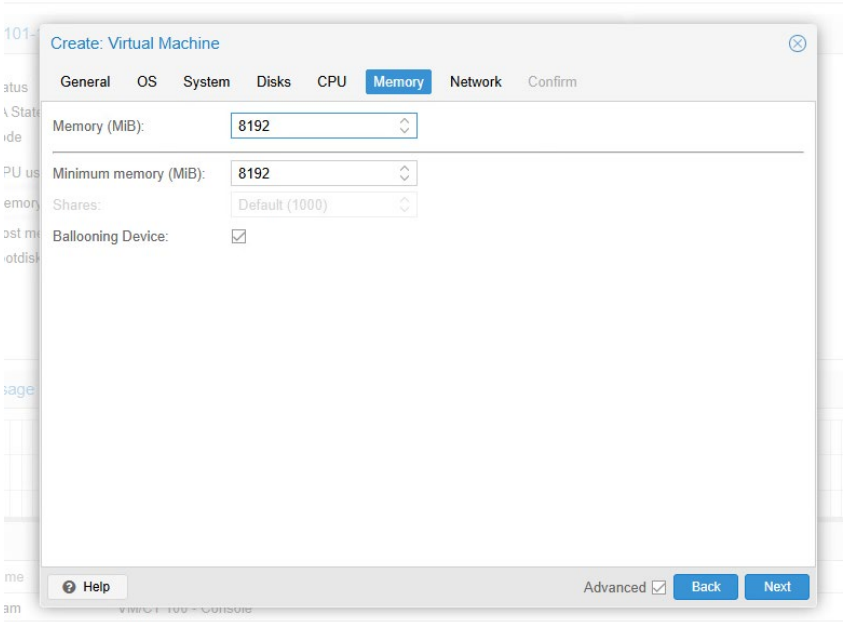
Step 25: Next

e.g.:



Step 26: Memory (MB): 8192

Step 27: Next



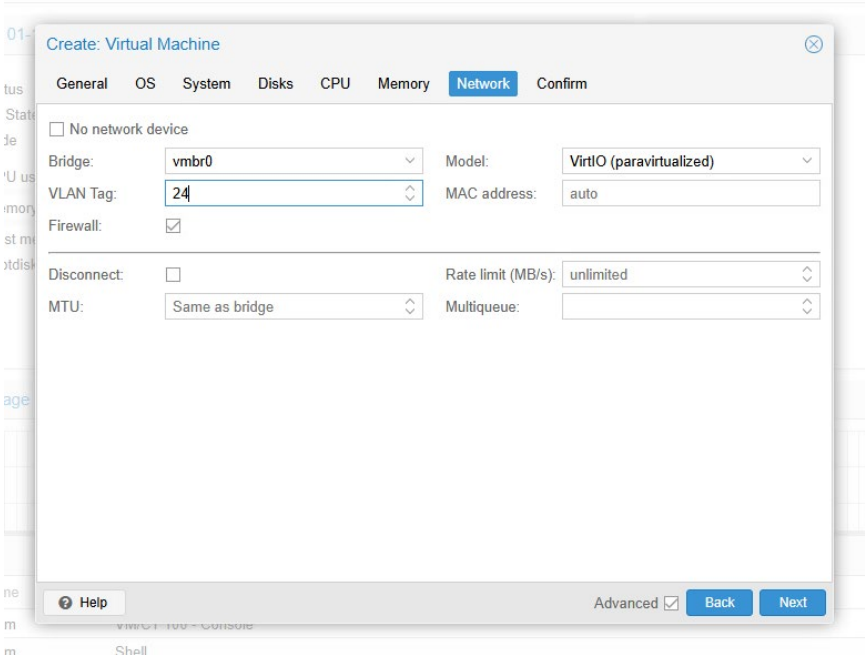
e.g.:

Step 28:

Step 29: Use defaults in most places *EXCEPT*

Step 30: VLAN Tag: 24

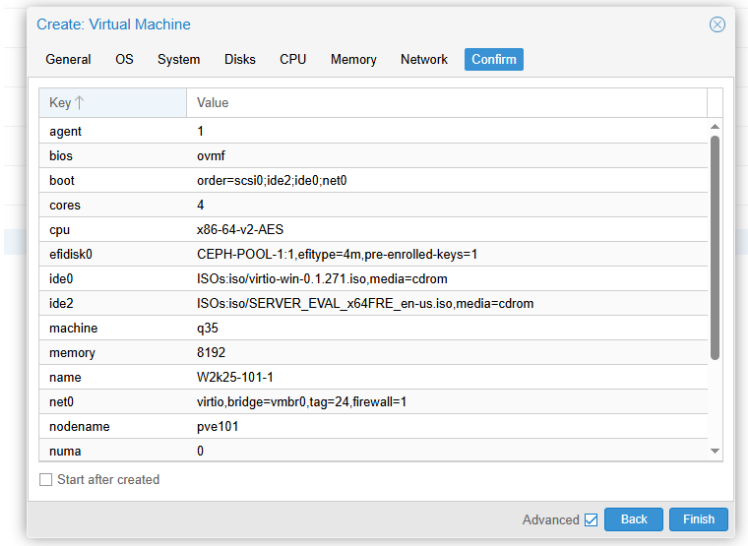
Step 31: Next



e.g.:

Step 32: Click finish

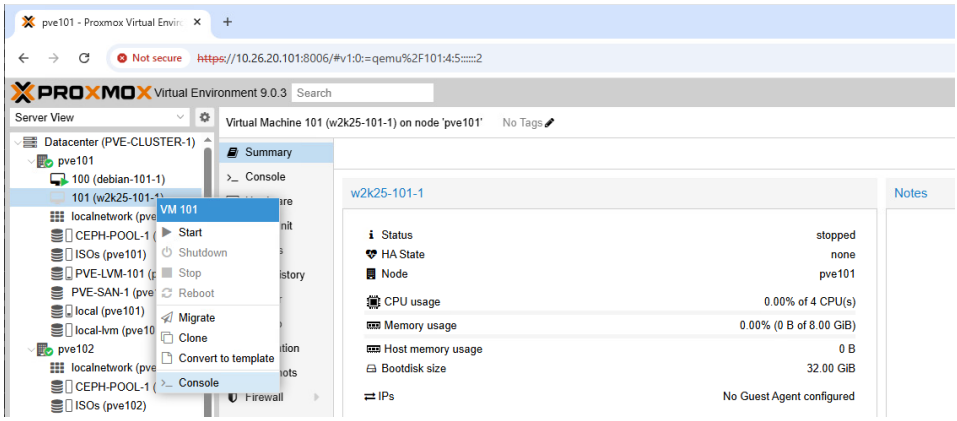
e.g.:



3. Wait for the VM to create

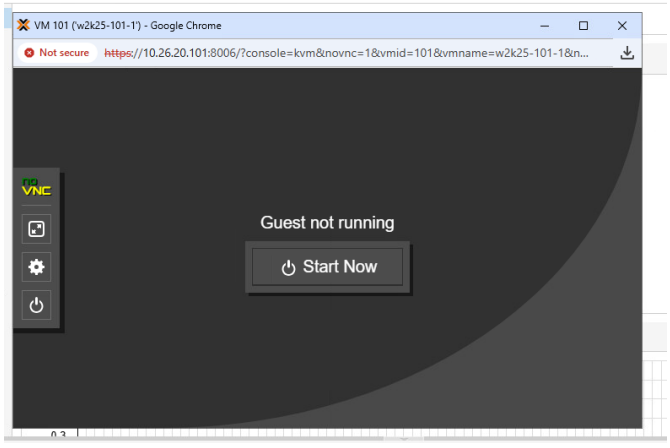
Step 1: Open a console to the VM

e.g.:



Step 2: Click: Start Now

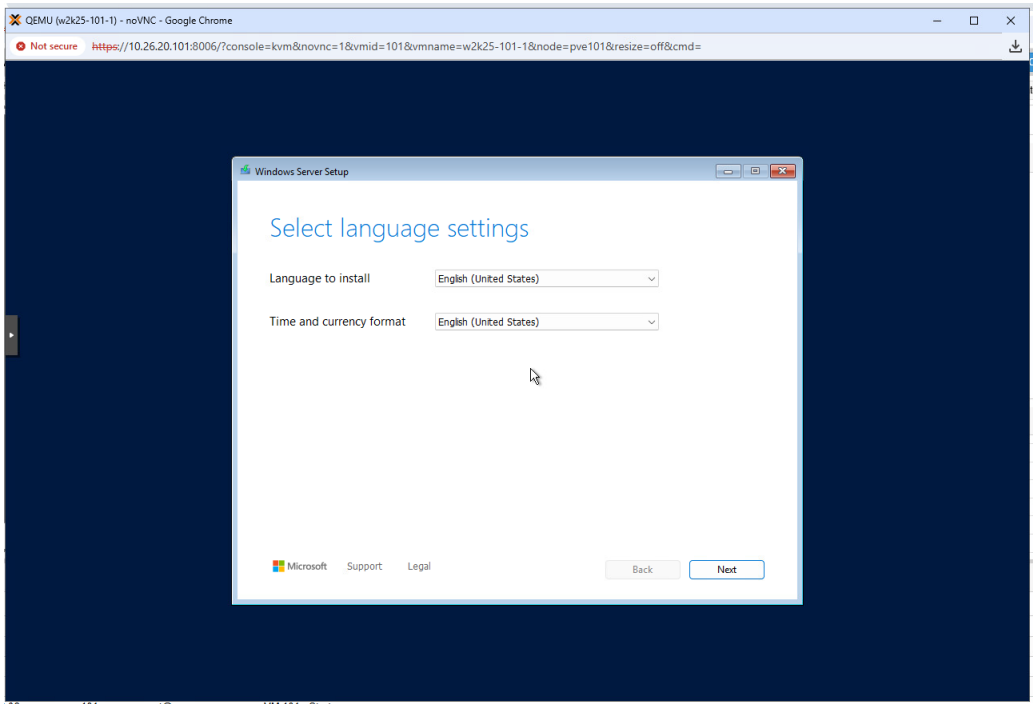
e.g.:

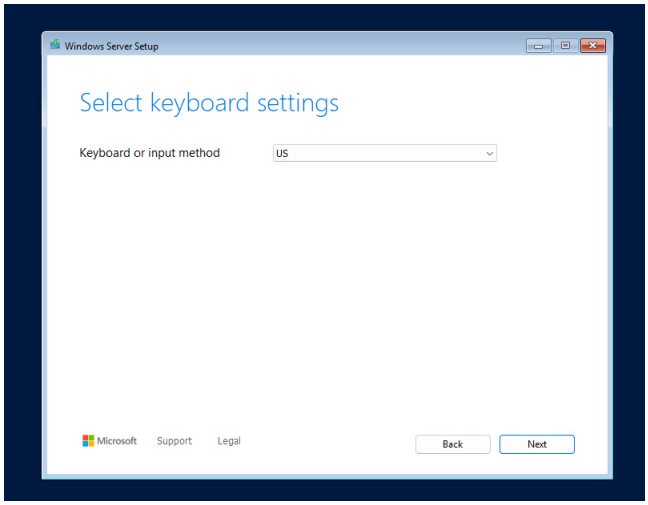


4. VM is now on

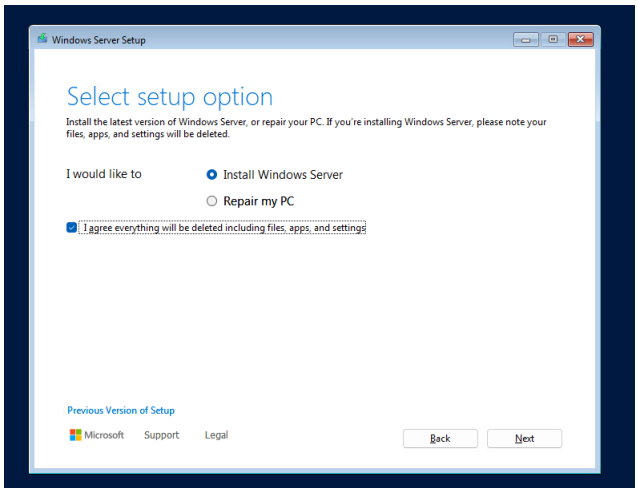
Step 1: Wait

e.g.: 

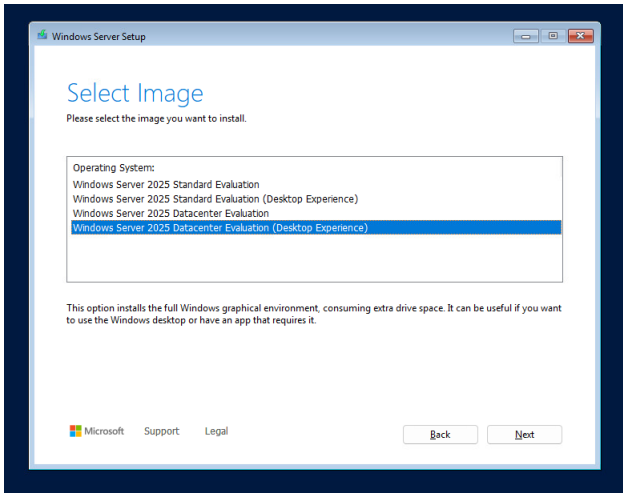
Step 2: 

Step 3: 

Step 4:

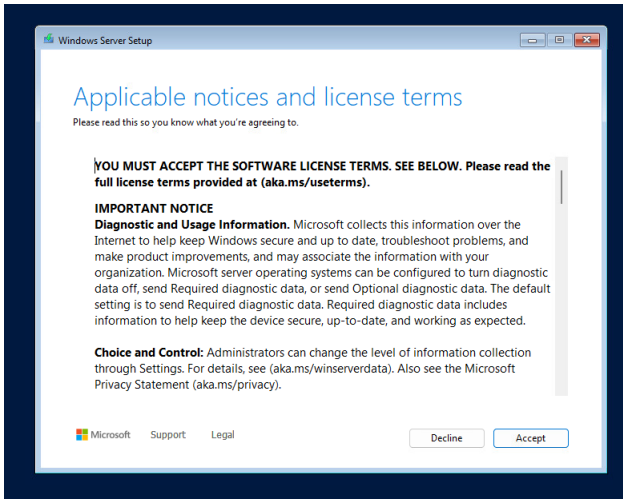


Step 5:



Step 6:

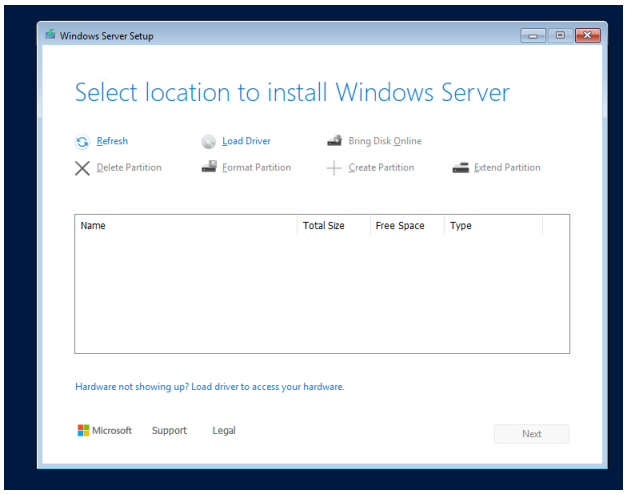
Step 7:



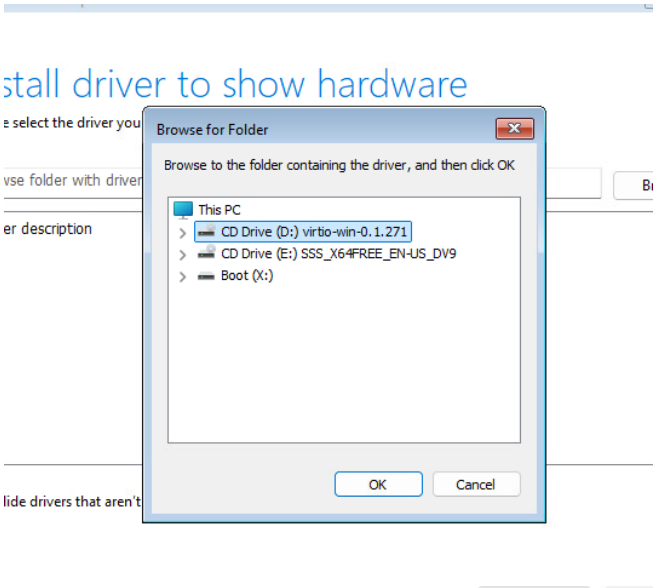
Step 8: Now we need to add virtio SCSI drivers to Windows in order to be able to see the disk

Step 9: Choose: Load Driver

e.g.:

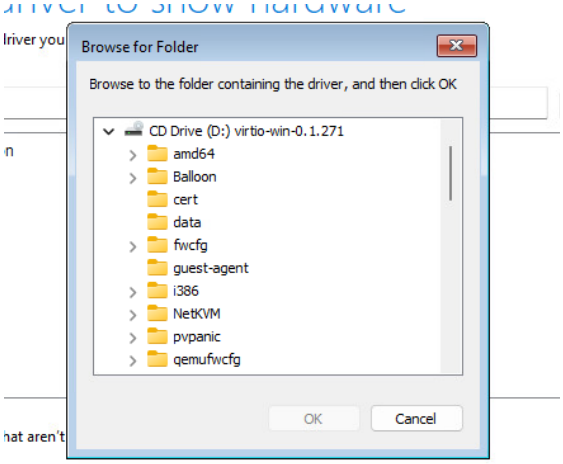


Step 10: Select the drive with virtio mounted and expand



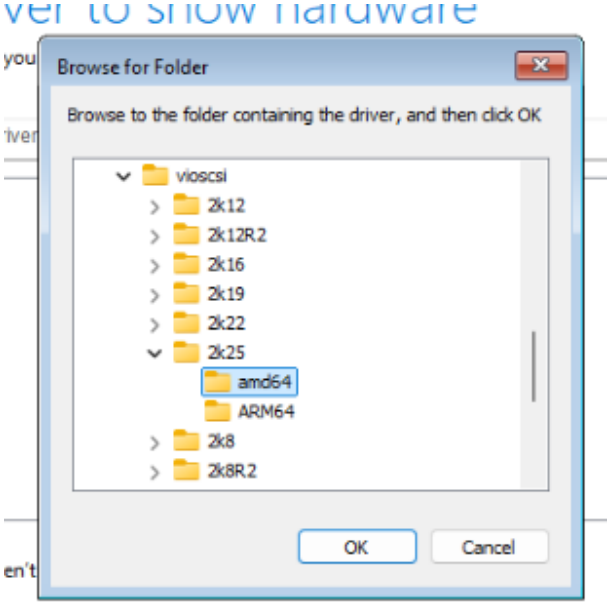
e.g.:

Step 11: Scroll down



e.g.:

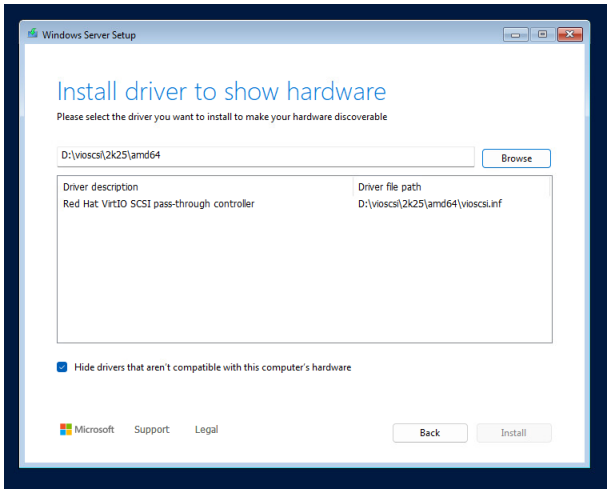
Step 12: Find the folder for your OS: vioscsi > 2k25 > amd64



e.g.:

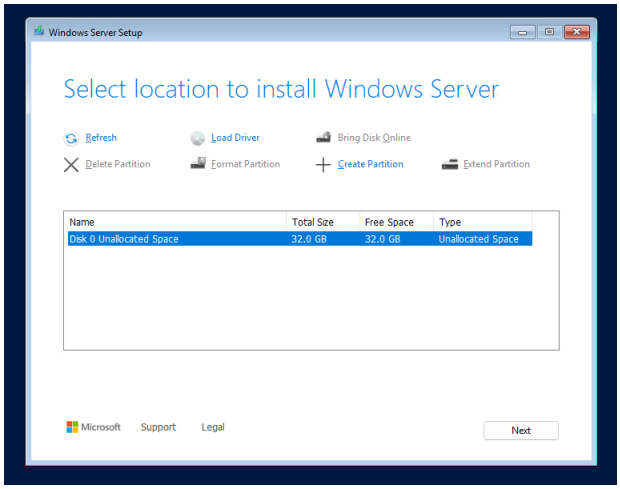
Step 13: Select the Red Hat VirtIO SCSI pass-through controller and click: Install

e.g.:



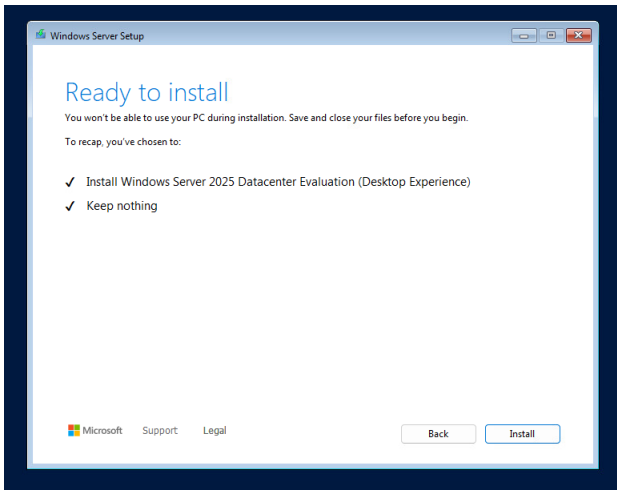
Step 14: Next

e.g.:

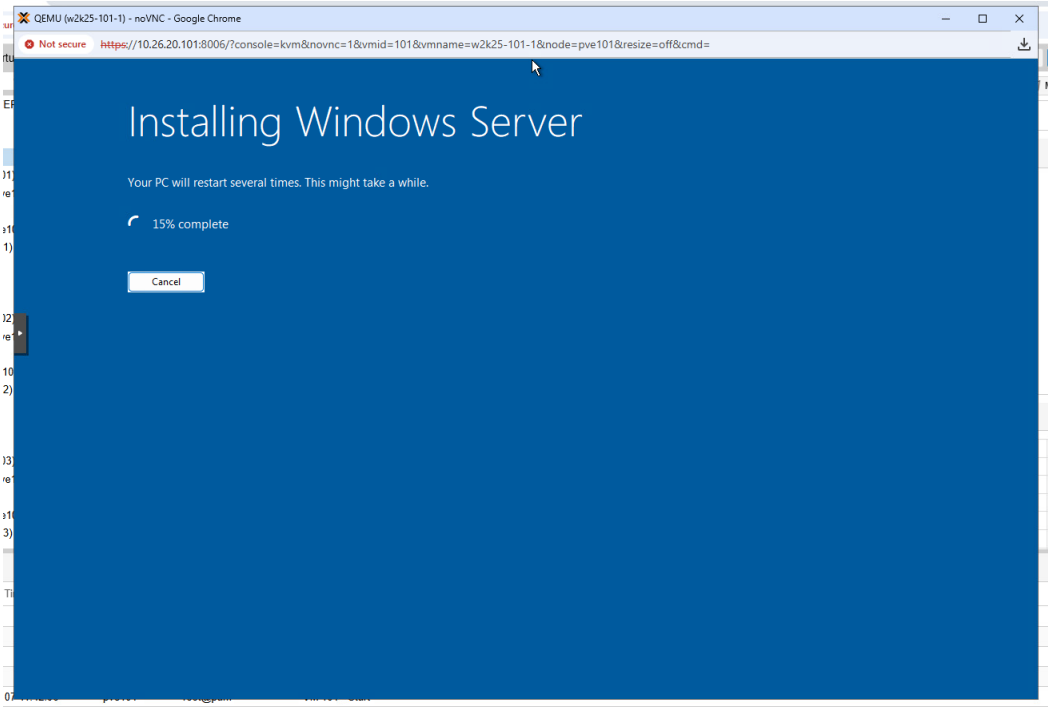


Step 15: Install

e.g.:

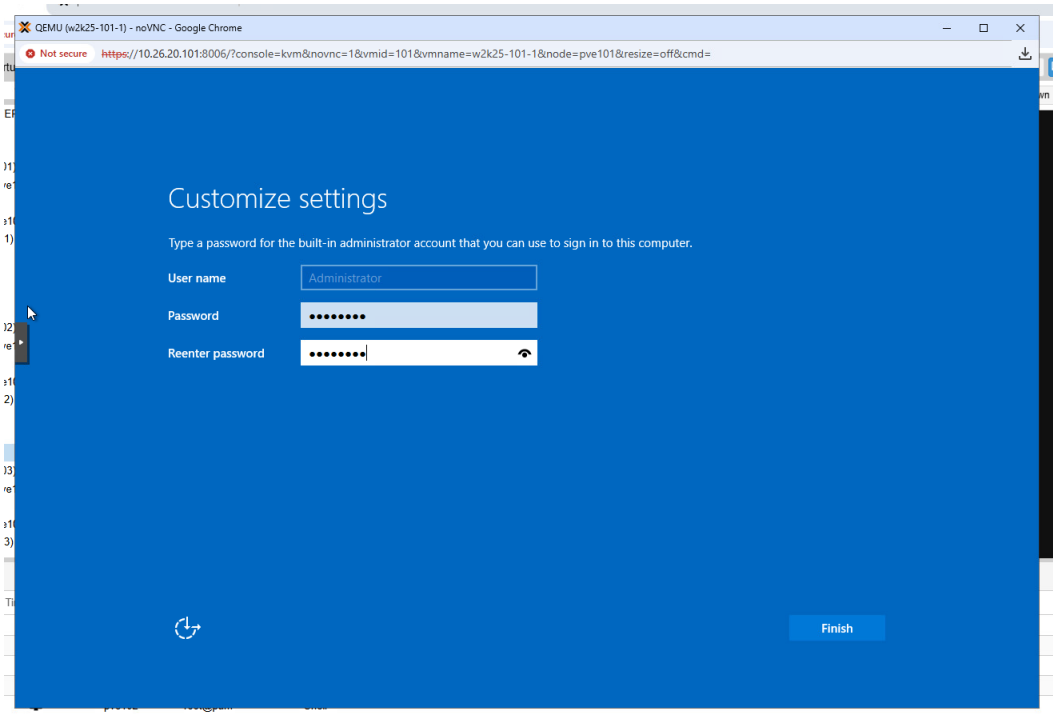


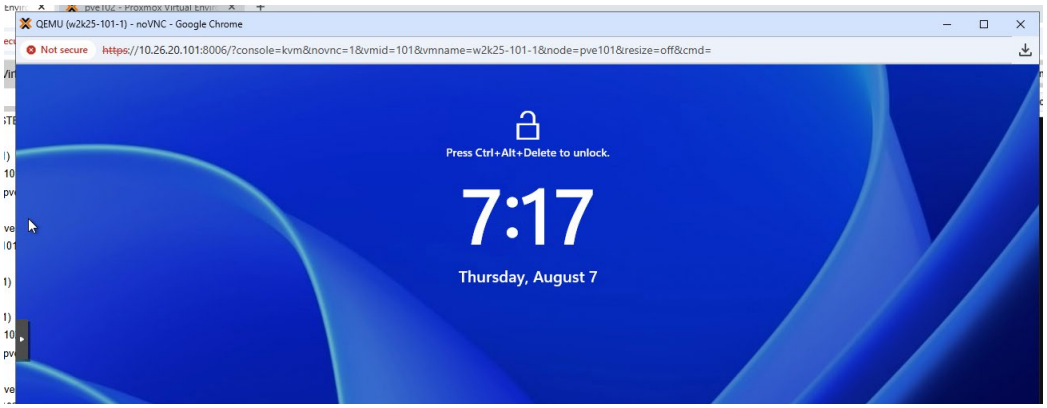
Step 16:



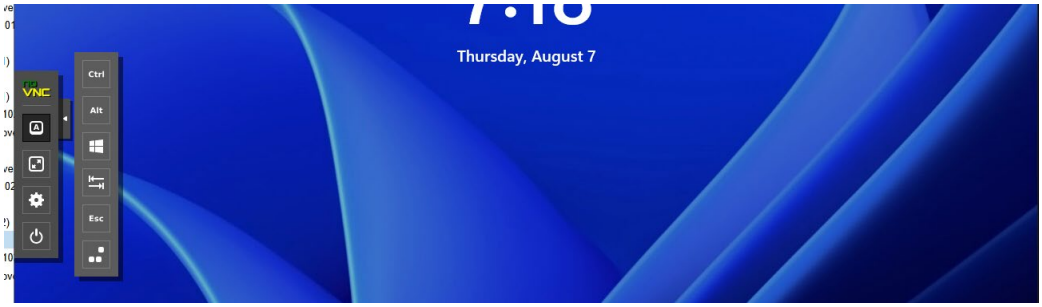
Step 17: Enter the password:

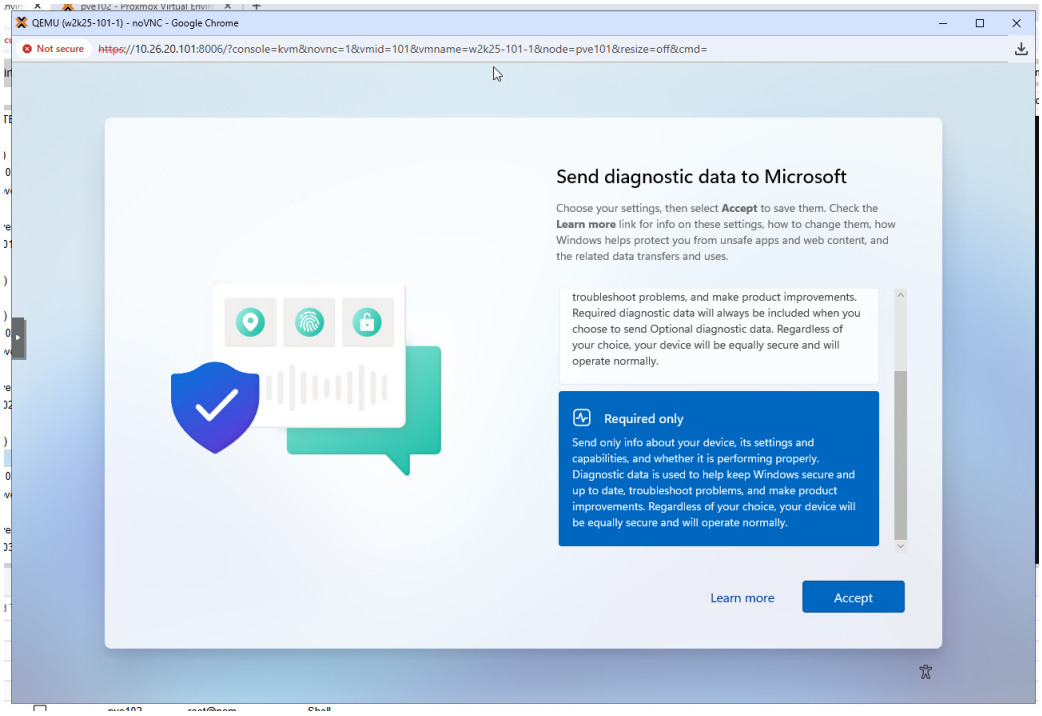
e.g.:



e.g.: 

Step 18: Use the floating keyboard to send: [CTRL]+[ALT]+[DEL]

e.g.: 

e.g.: 

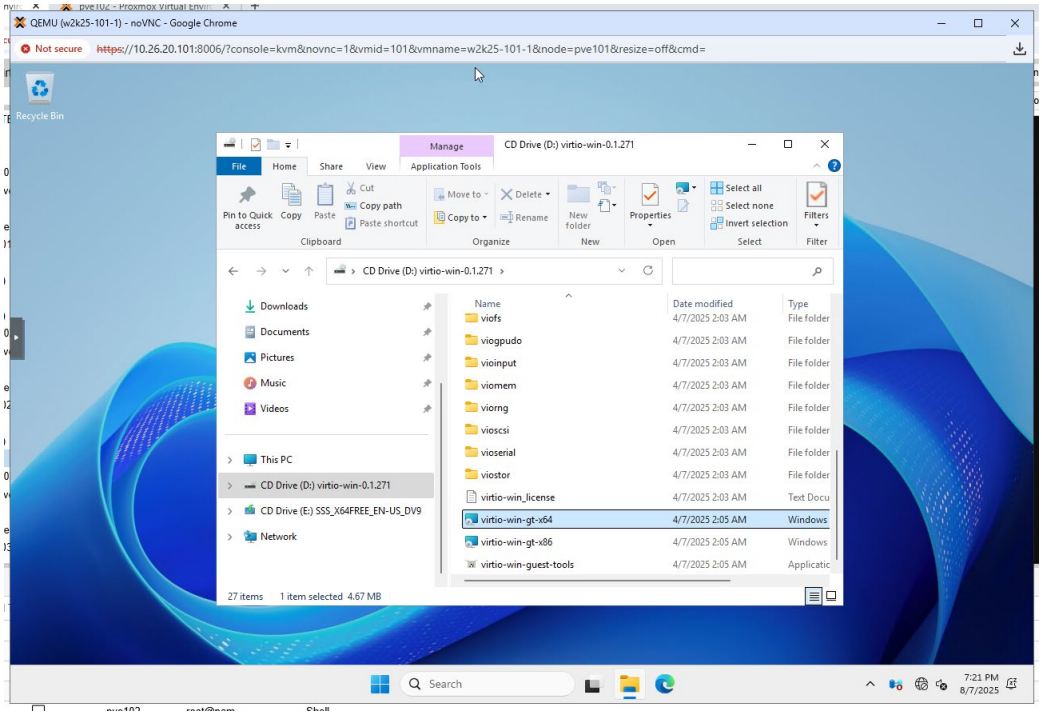
e.g.:

e.g.:

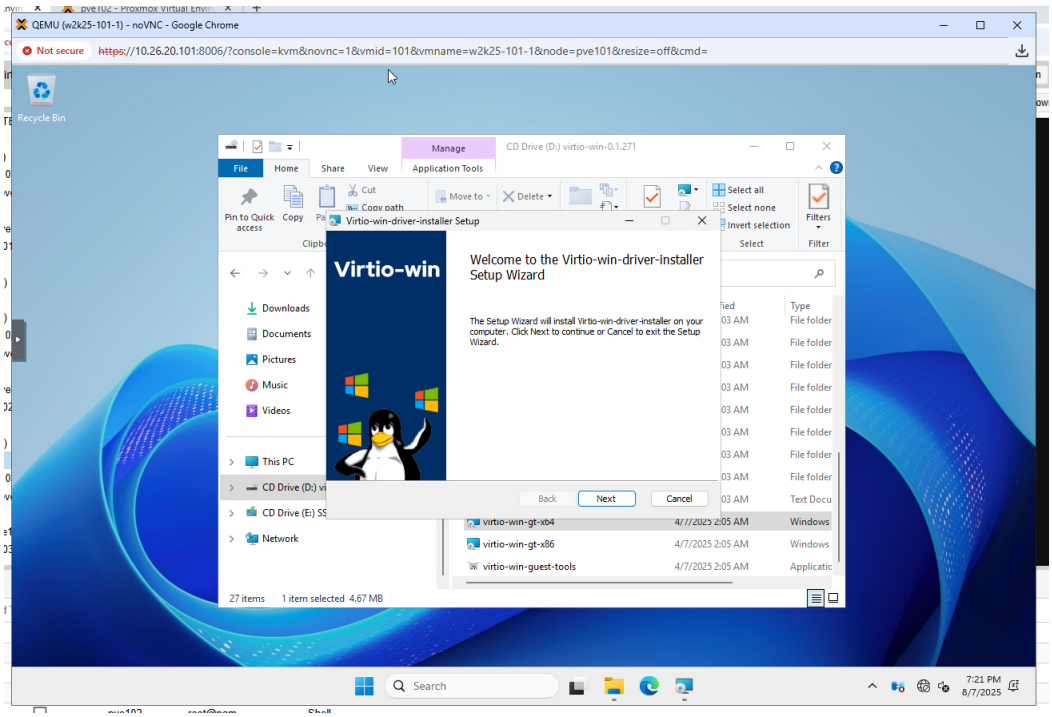
SBS LAB – (GUI) VirtIO for Windows with QEMU Guest Agent

Step 1: Open the file browser to the virtio drive and run: virtio-win-gt-x64

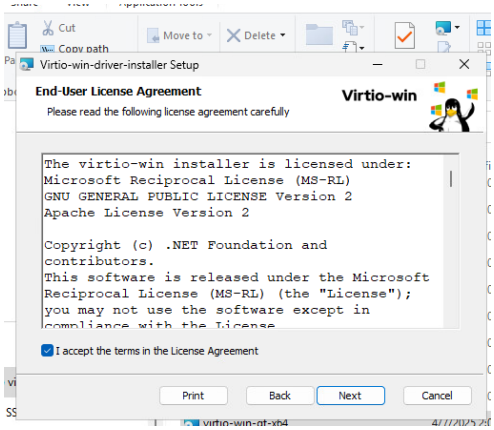
e.g.:



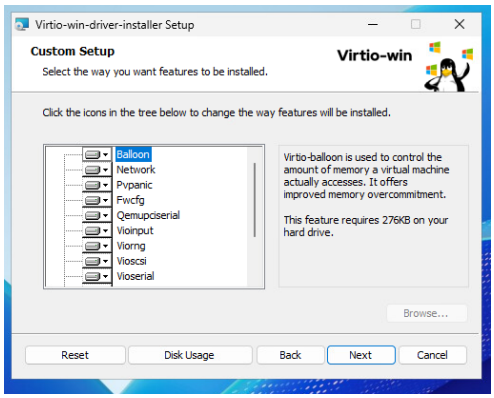
Step 2:



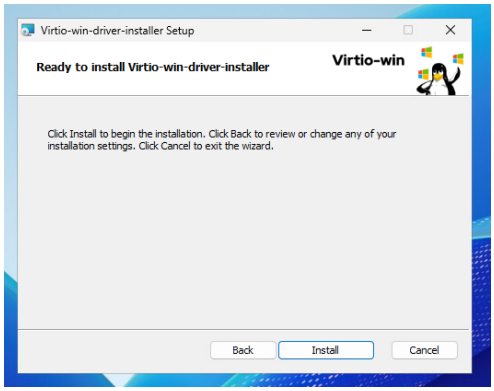
Step 3:



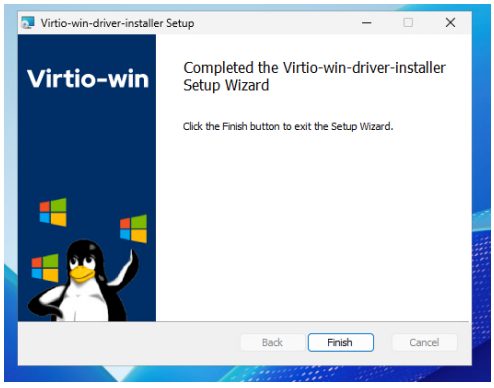
Step 4:



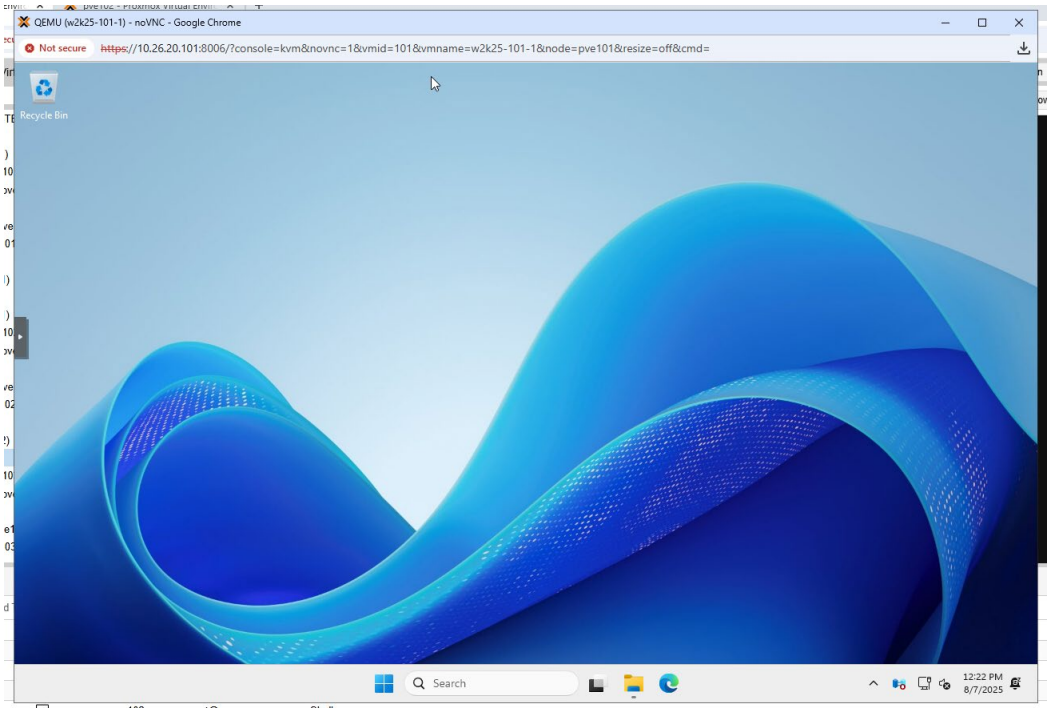
Step 5:



Step 6:

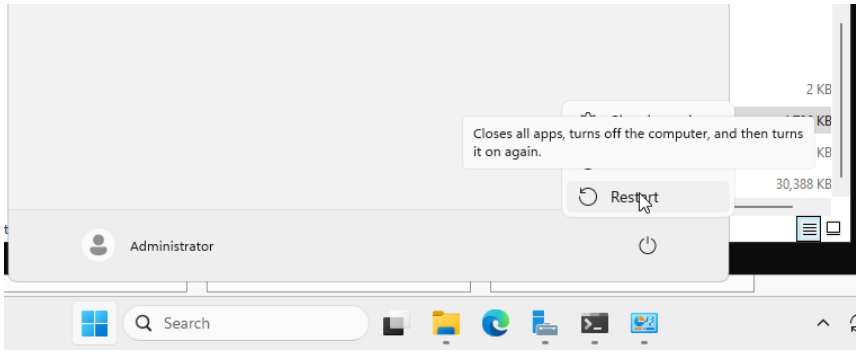


e.g.:



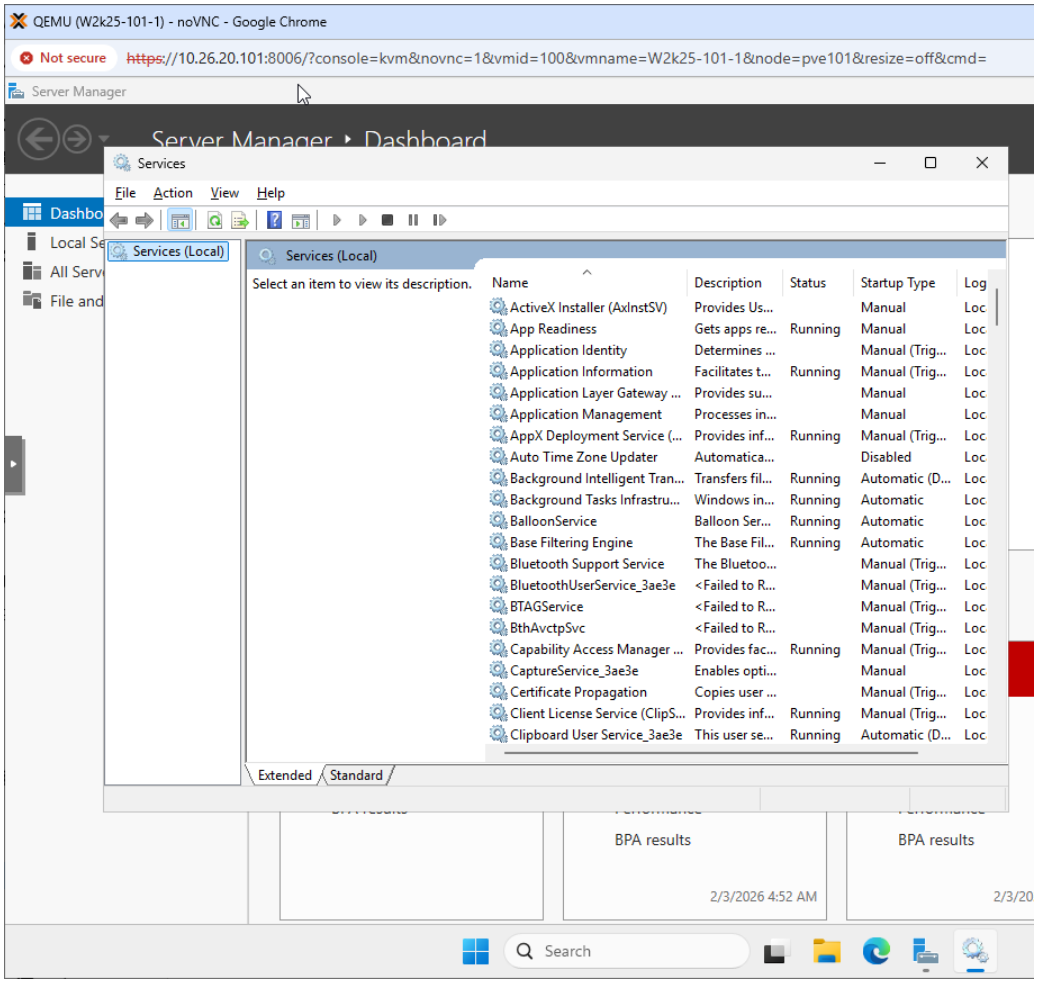
Step 7: Reboot Windows

e.g.:



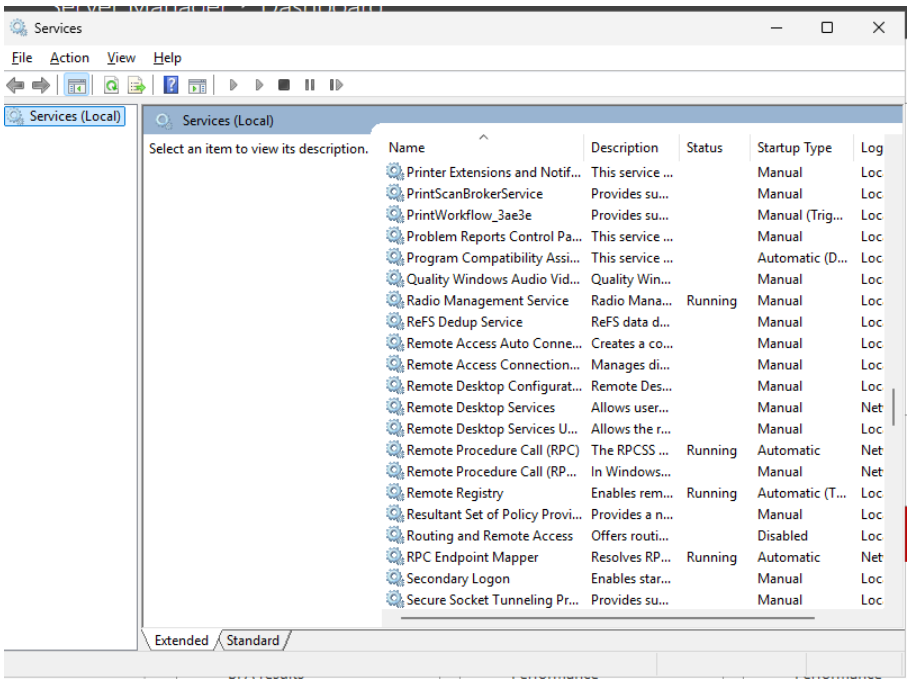
Step 8: Login to w2k25-XYZ-1 > Search: services.msc > [Enter]

e.g.:



Step 9: Look for the QEMU Guest Agent

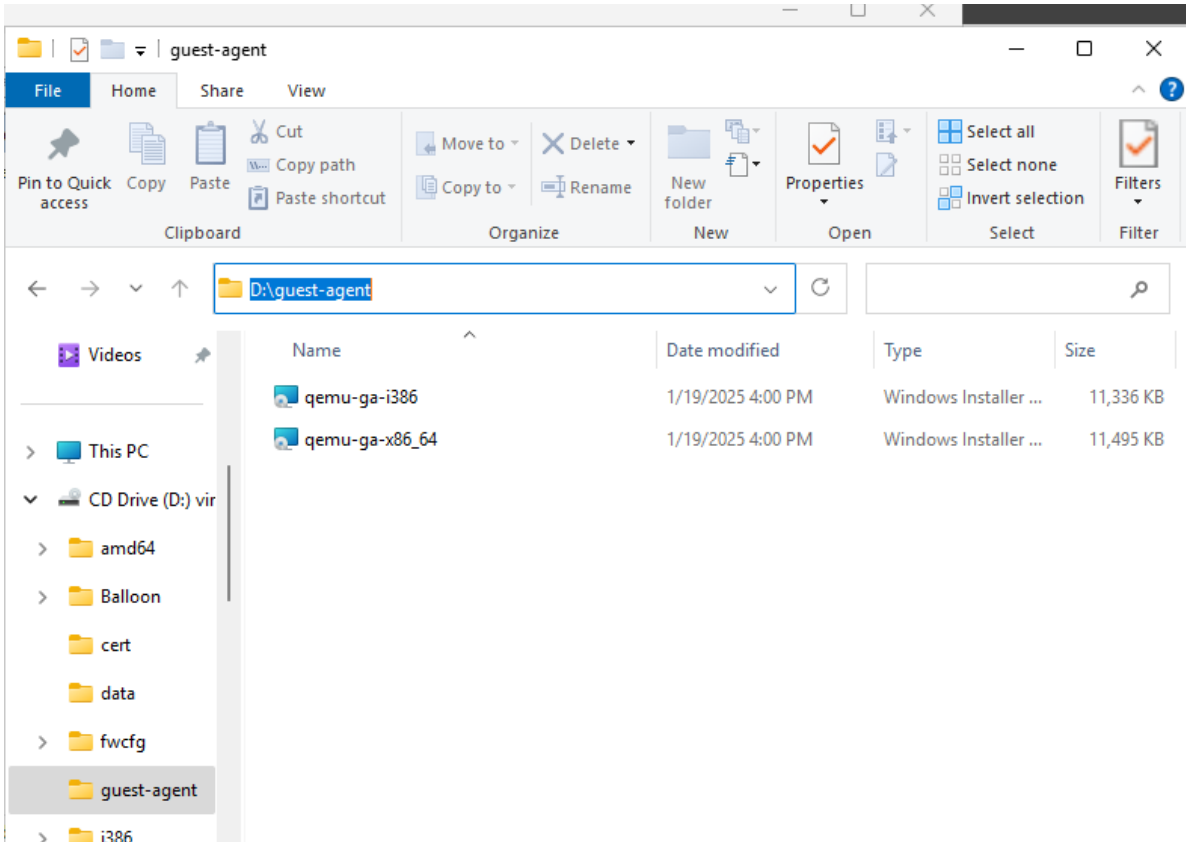
e.g.:



Step 10: If it is not present, you will need to install it manually from CD/DVD

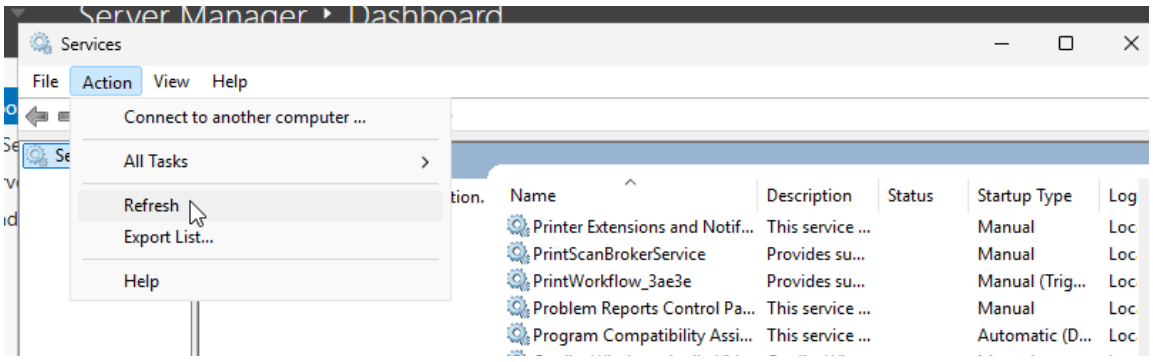
Step 11: Install QEMU Guest Agent by browsing to: D:\ virtio-win-X.X.XXX\guest-agent\qemu-ga-x86_64 and double-click

e.g.:

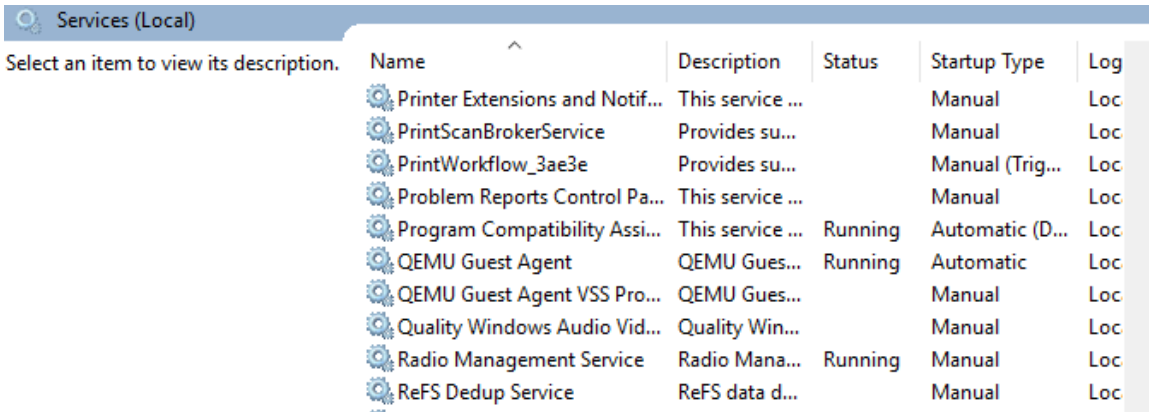


Step 12: Check services.msc again after: Refresh

e.g.:



e.g.:


Services (Local)
Select an item to view its description.
Name
Description
Status
Startup Type
Log
Printer Extensions and Notif...
This service ...
Manual
Loc
PrintScanBrokerService
Provides su...
Manual
Loc
PrintWorkflow_3ae3e
Provides su...
Manual (Trig...
Loc
Problem Reports Control Pa...
This service ...
Manual
Loc
Program Compatibility Assi...
This service ...
Running
Automatic (D...
Loc
QEMU Guest Agent
QEMU Gues...
Running
Automatic
Loc
QEMU Guest Agent VSS Pro...
QEMU Gues...
Manual
Loc
Quality Windows Audio Vid...
Quality Win...
Manual
Loc
Radio Management Service
Radio Mana...
Running
Manual
Loc
ReFS Dedup Service
ReFS data d...
Manual
Loc

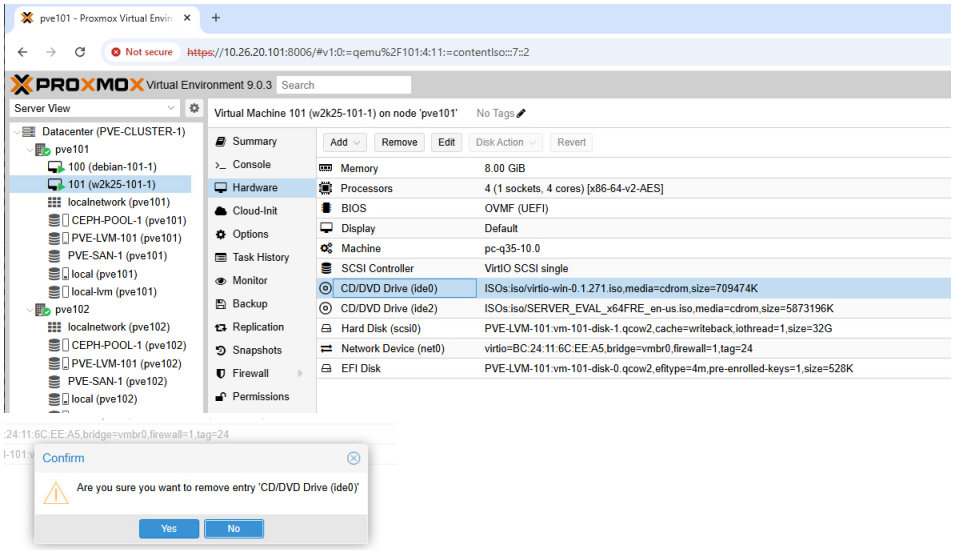
SBS LAB –(GUI) VM Cleanup after installation

Step 1: Go to: pveXYZ > w2k25-XYZ-1 > Hardware

Step 2: Select the first CD/DVD

Step 3: Remove

e.g.:

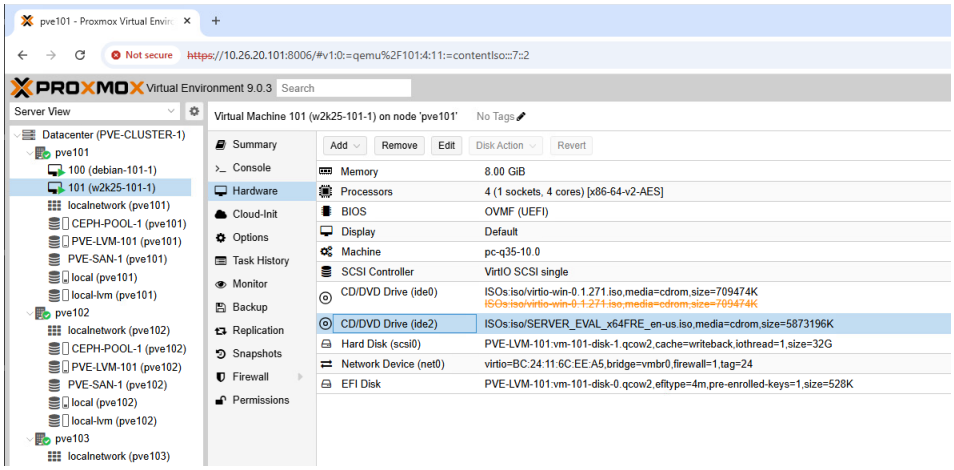


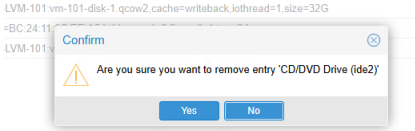
e.g.:

Step 4: Select the second CD/DVD

Step 5: Remove

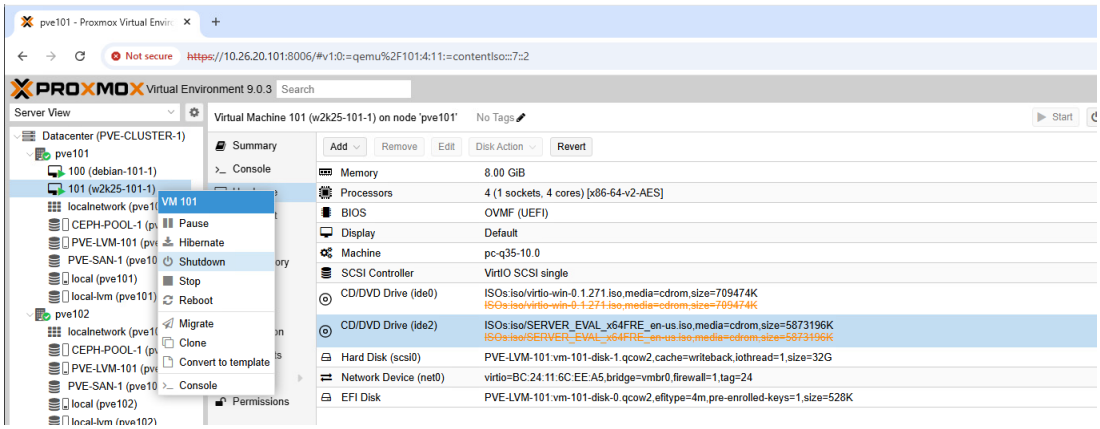
e.g.:





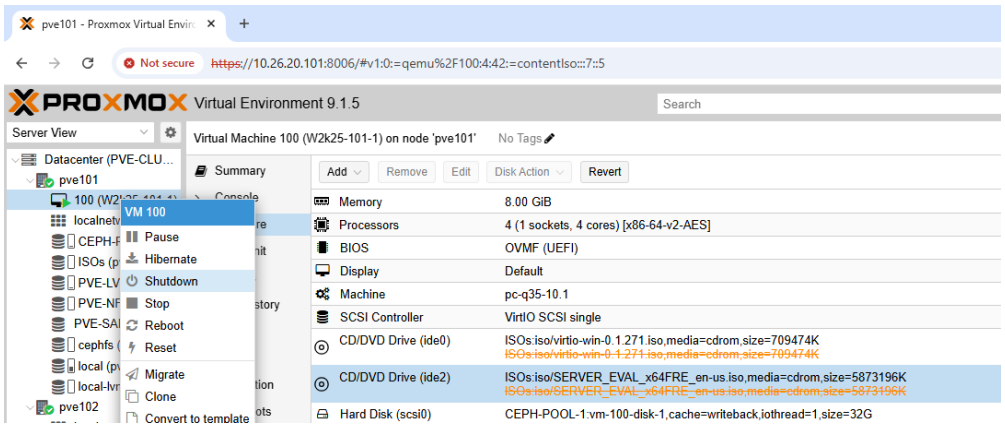
e.g.:

e.g.:

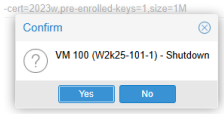


Step 6: Right-click on the VM > Shutdown

e.g.:



e.g.:



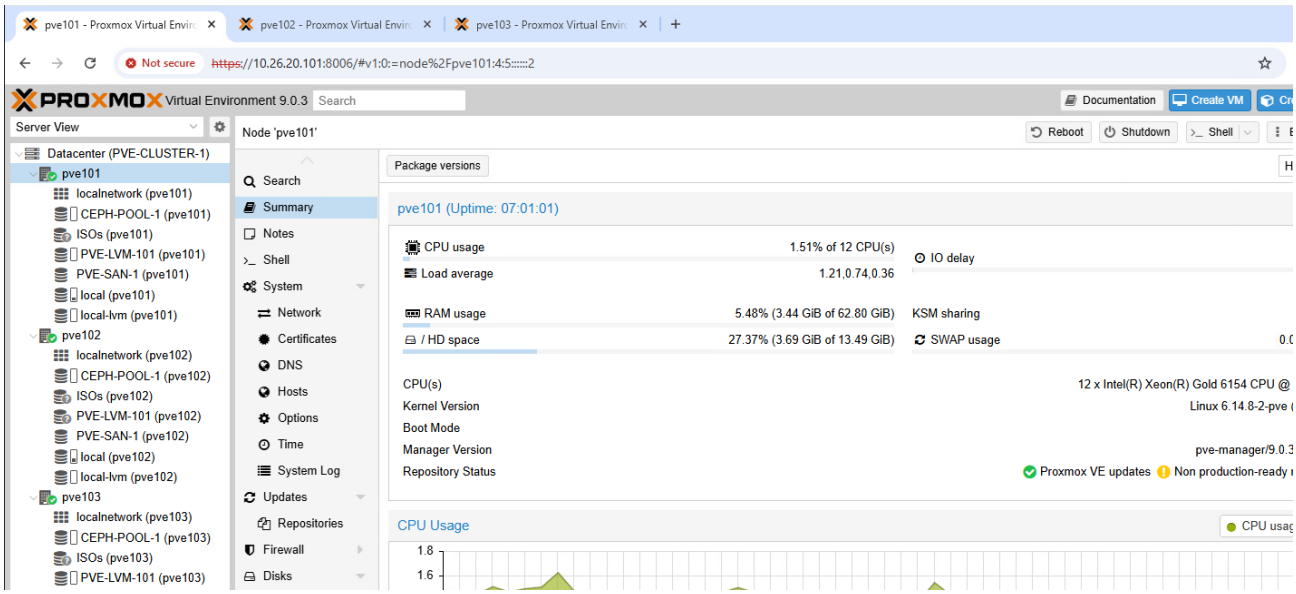
NOTE : The devices will only be fully removed after VM shutdown

SBS LAB – (GUI/CLI) Debian Linux Server for PVE

1. The first thing we are going to do is build the VM

Step 1: Create VM

e.g.:

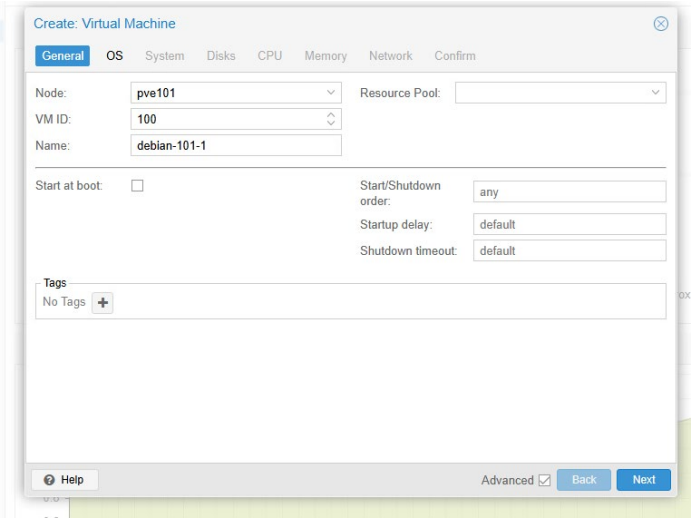


Step 2: Choose node: pveXYZ

Step 3: Name: debian-ZYZ-1

Step 4: Next

e.g.:

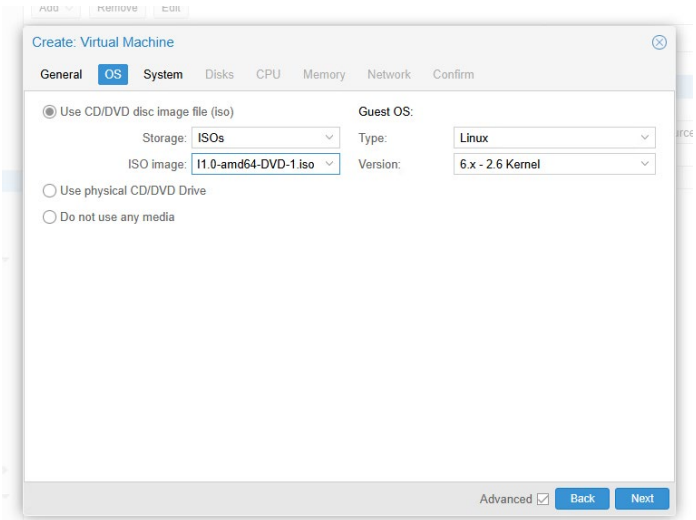


Step 5: Storage: ISOs

Step 6: ISO Image: Debian-...

Step 7: Next

e.g.:

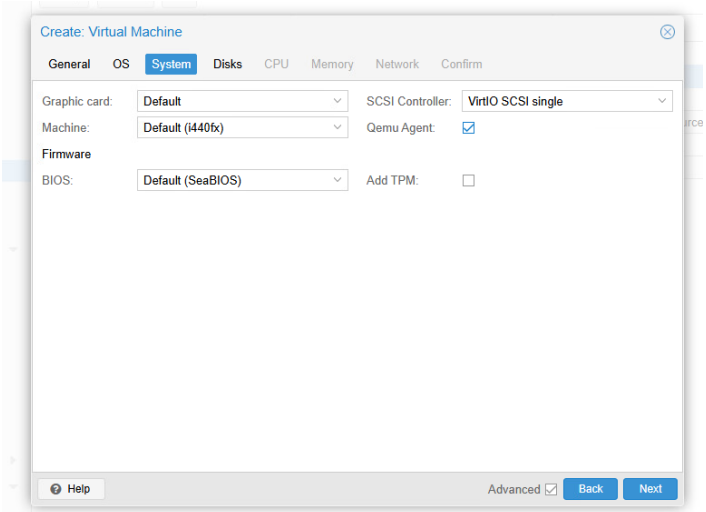


Step 8: Use defaults in most places *EXCEPT*

Step 9: Qemu Agent: checked

Step 10: Next

e.g.:



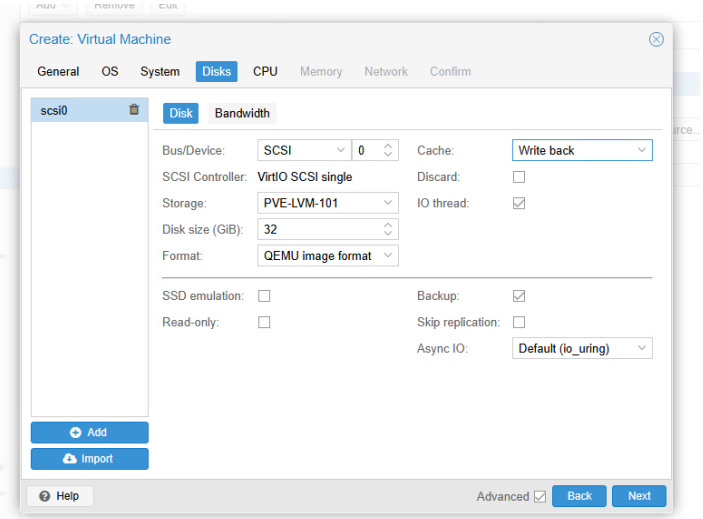
Step 11: Use defaults in most places *EXCEPT*

Step 12: Cache: Write back (fast and safe)

Step 13: Storage: PVE-LVM-XYZ

Step 14: Next

e.g.:



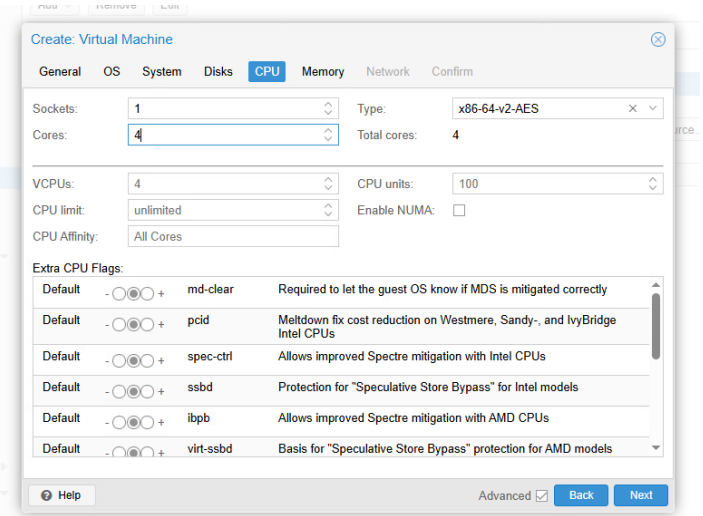
Step 15: Use defaults in most places *EXCEPT*

Step 16: Sockets: 1

Step 17: Cores: 4

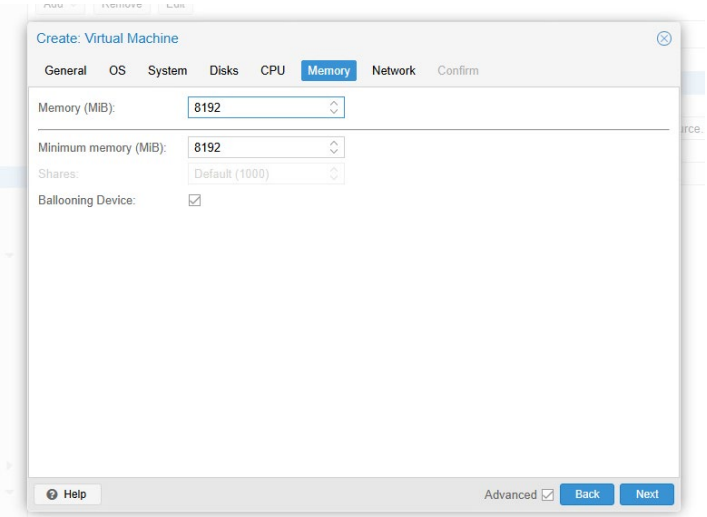
Step 18: Next

e.g.:



Step 19: Memory (MiB): 8192

Step 20: Next



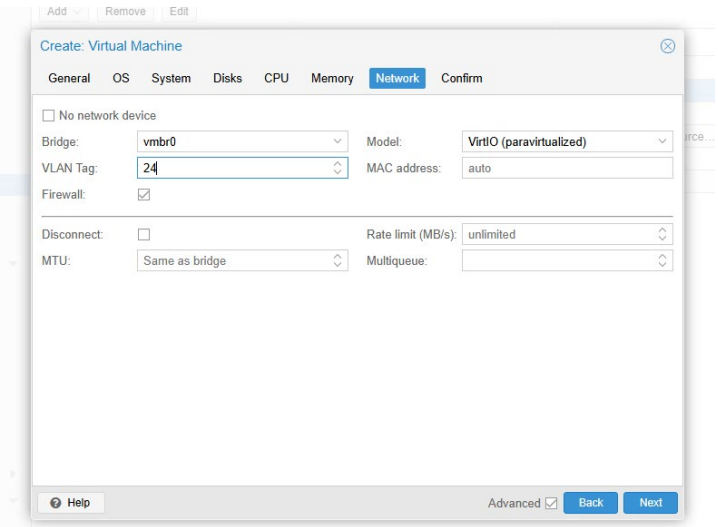
e.g.:

Step 21: Use defaults in most places *EXCEPT*

Step 22: Bridge: vmbr0

Step 23: VLAN Tag: 24

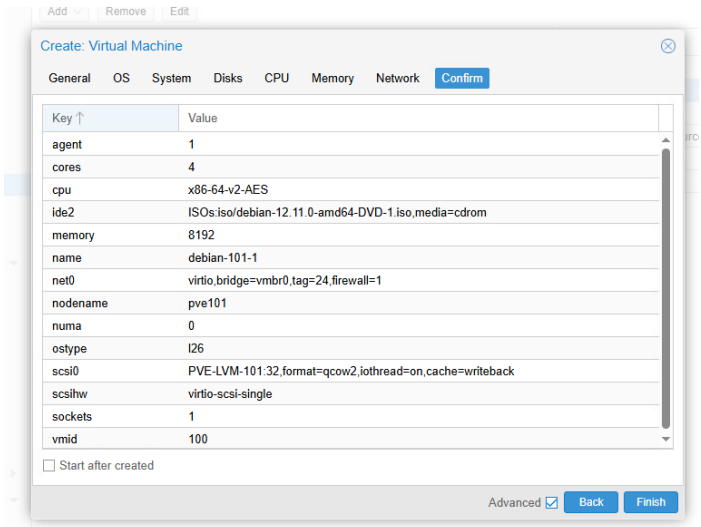
Step 24: Next



e.g.:

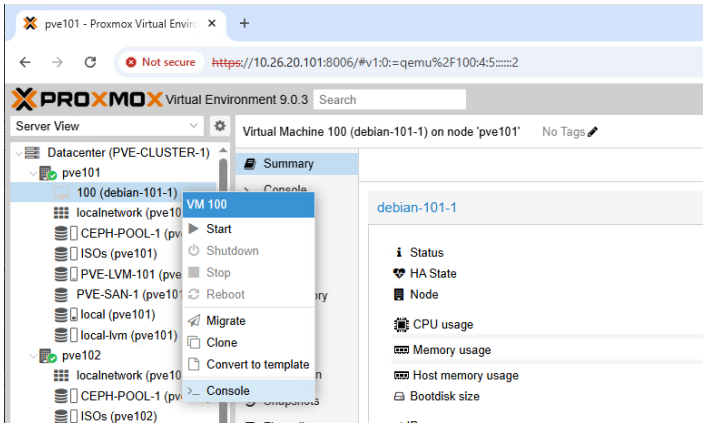
Step 25: Finish

e.g.:



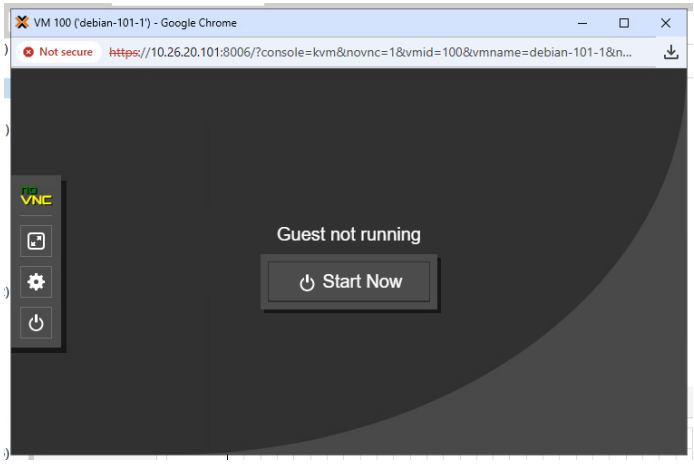
Step 26: Right-click to: Console

e.g.:



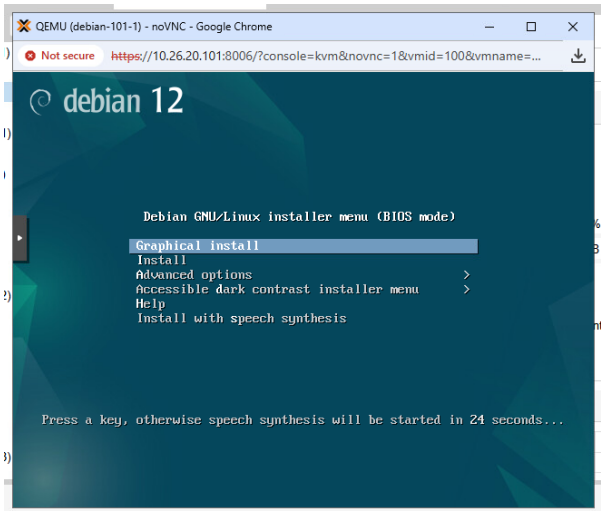
Step 27: Start

e.g.:

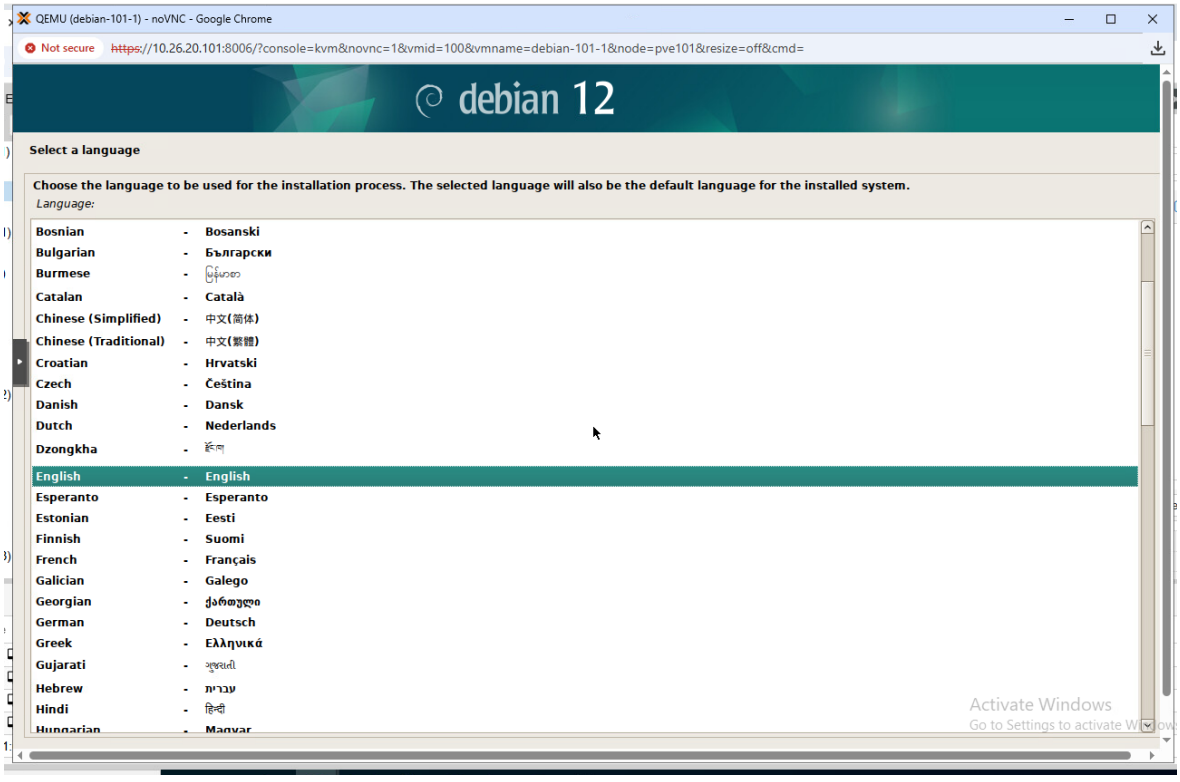


2. Now we install Debian

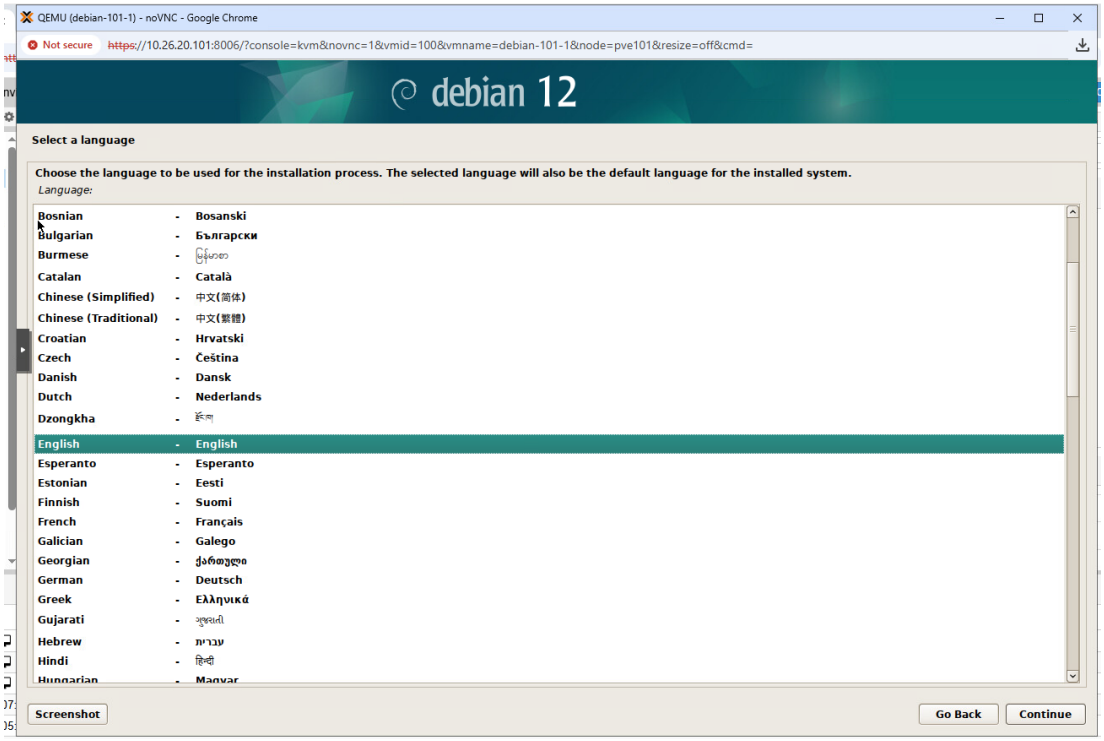
Step 1:



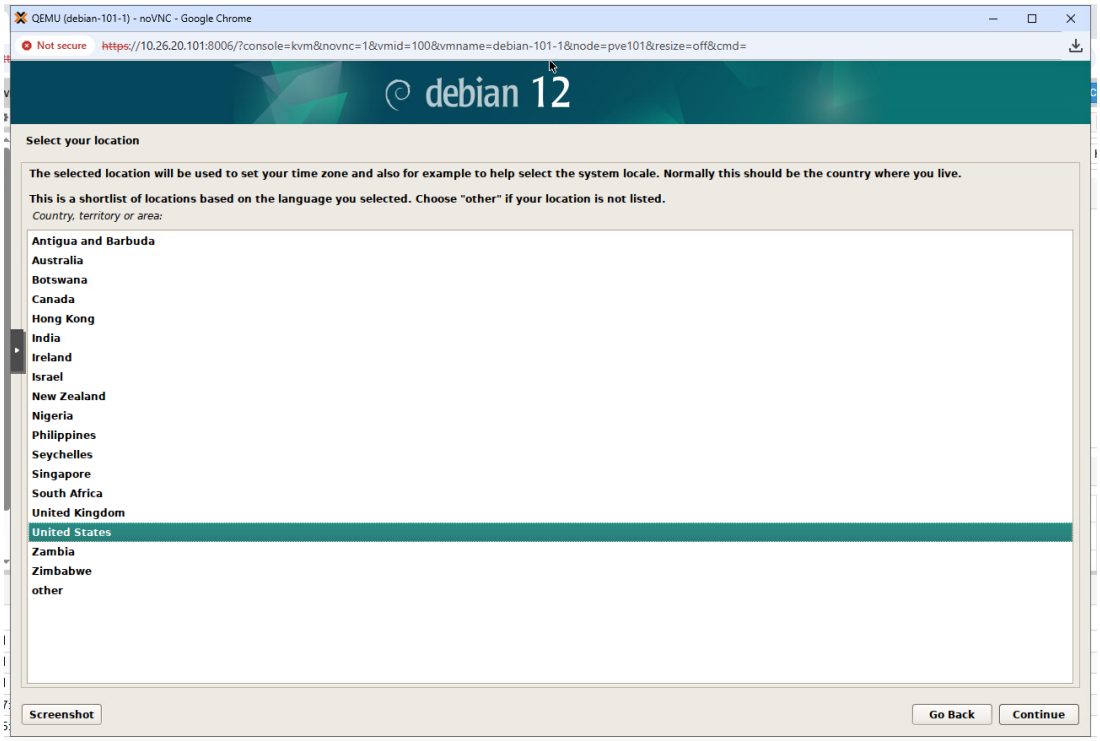
Step 2:



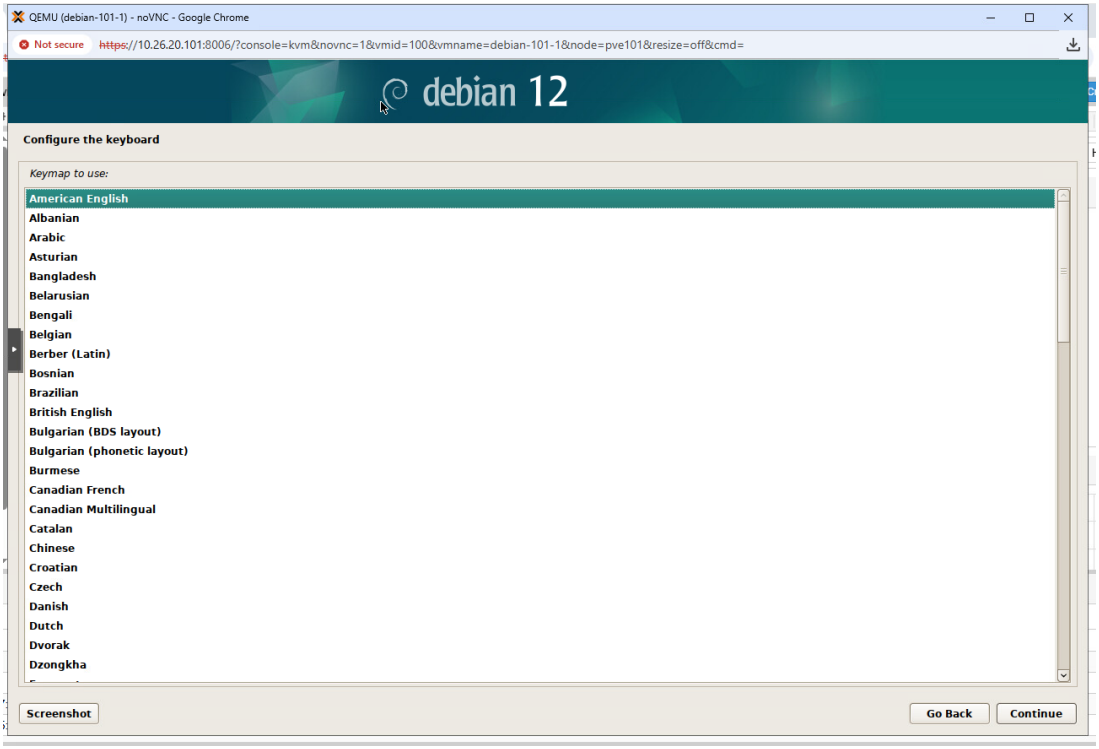
Step 3:



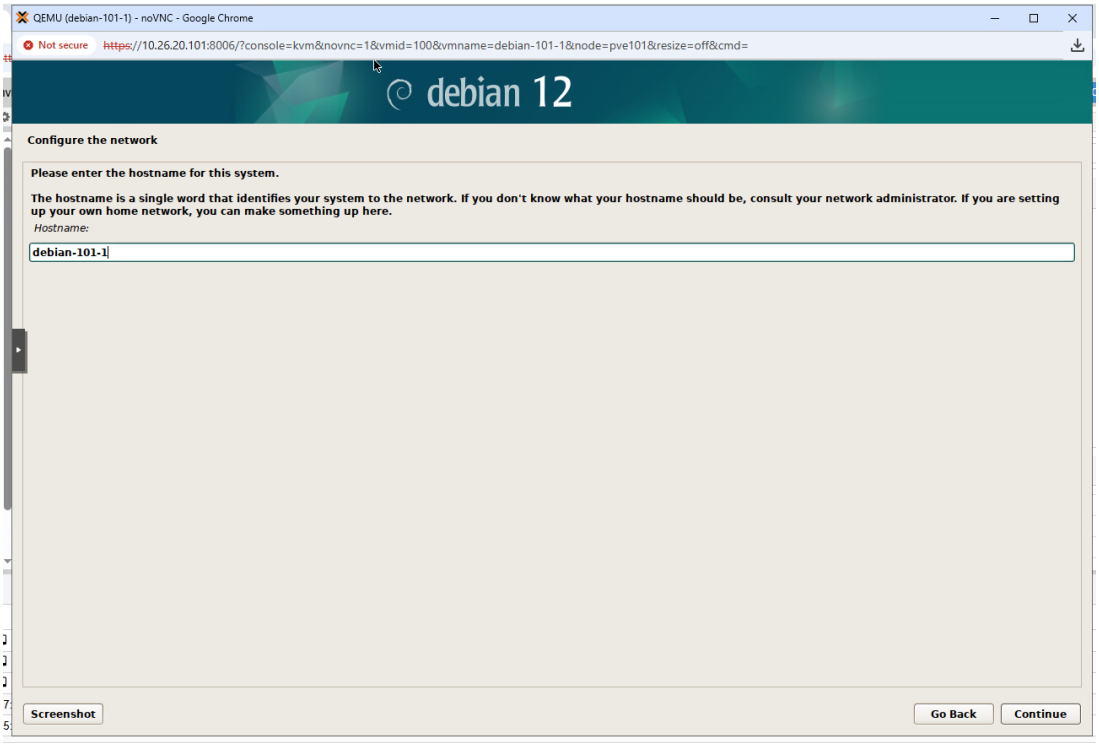
Step 4:



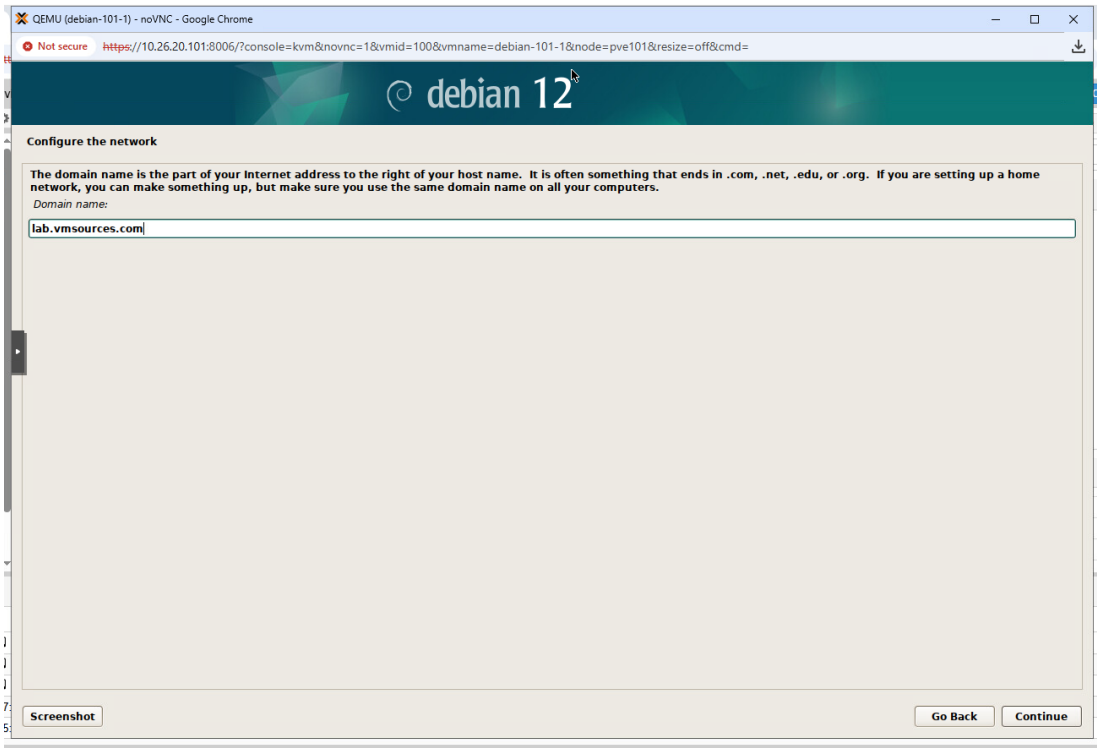
Step 5:


QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
debian 12
Configure the keyboard
Keymap to use:
American English
Albanian
Arabic
Asturian
Bangladesh
Belarusian
Bengali
Belgian
Berber (Latin)
Bosnian
Brazilian
British English
Bulgarian (BDS layout)
Bulgarian (phonetic layout)
Burmese
Canadian French
Canadian Multilingual
Catalan
Chinese
Croatian
Czech
Danish
Dutch
Dvorak
Dzongkha
Screenshot
Go Back
Continue

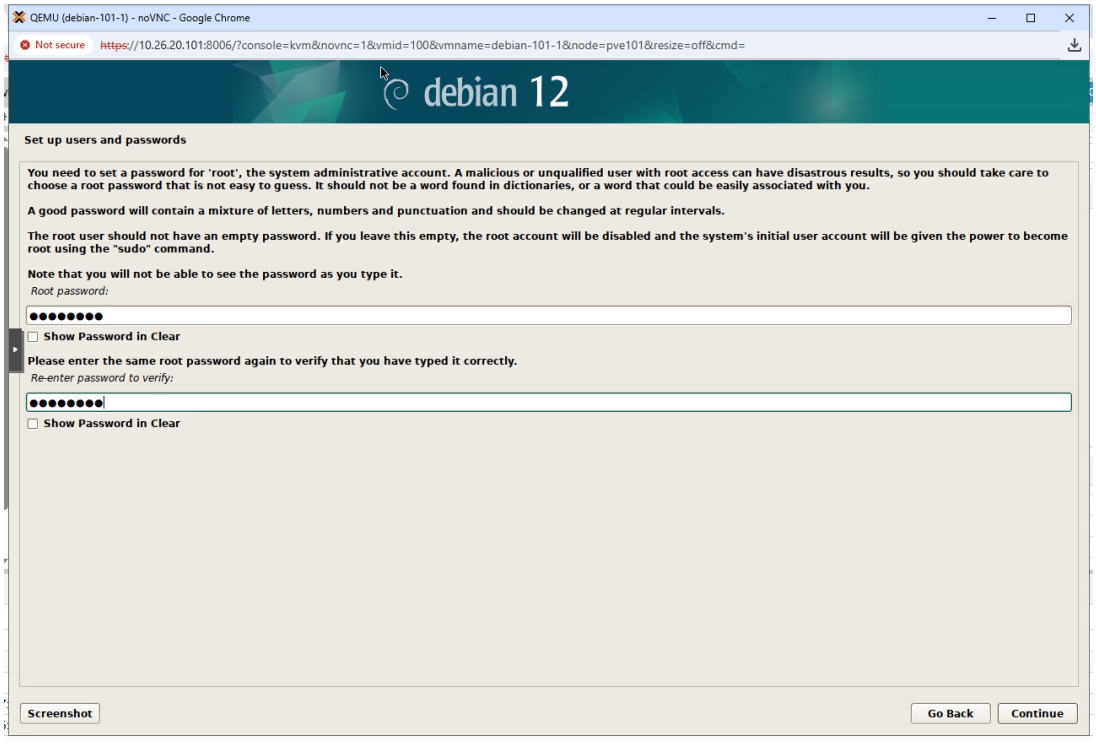
Step 6:


QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
debian 12
Configure the network
Please enter the hostname for this system.
The hostname is a single word that identifies your system to the network. If you don't know what your hostname should be, consult your network administrator. If you are setting up your own home network, you can make something up here.
Hostname:
debian-101-1
Screenshot
Go Back
Continue

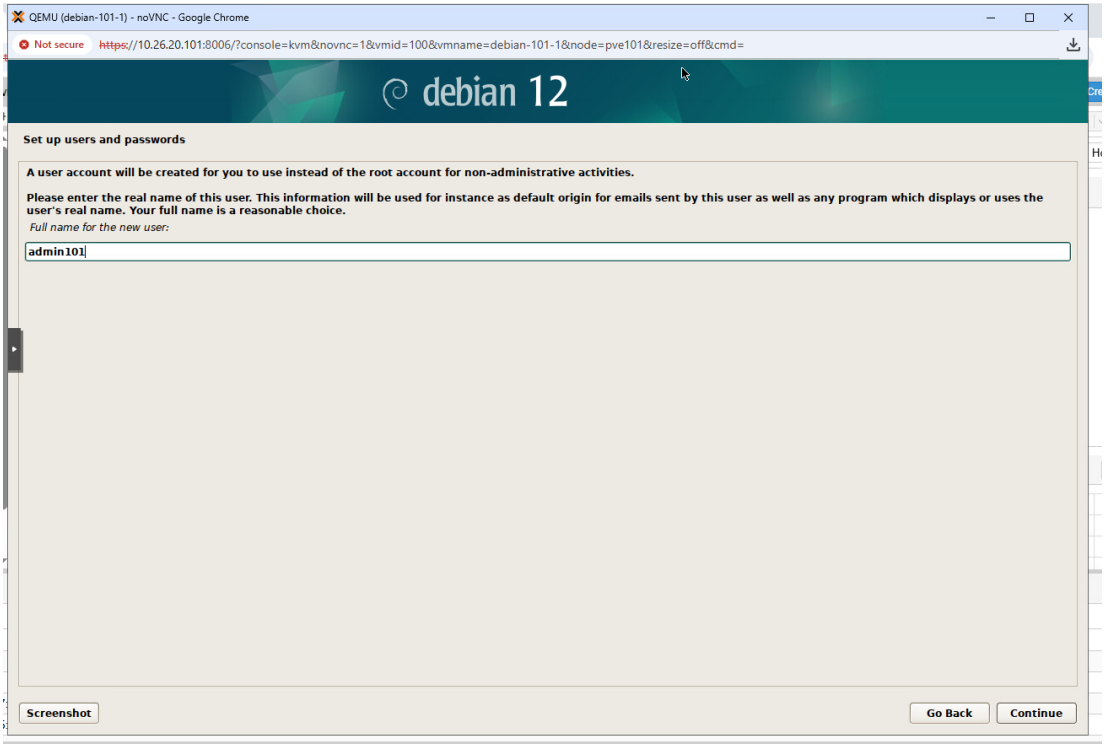
Step 7:


QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
debian 12
Configure the network
The domain name is the part of your Internet address to the right of your host name. It is often something that ends in .com, .net, .edu, or .org. If you are setting up a home network, you can make something up, but make sure you use the same domain name on all your computers.
Domain name:
lab.vmsources.com
Screenshot
Go Back
Continue

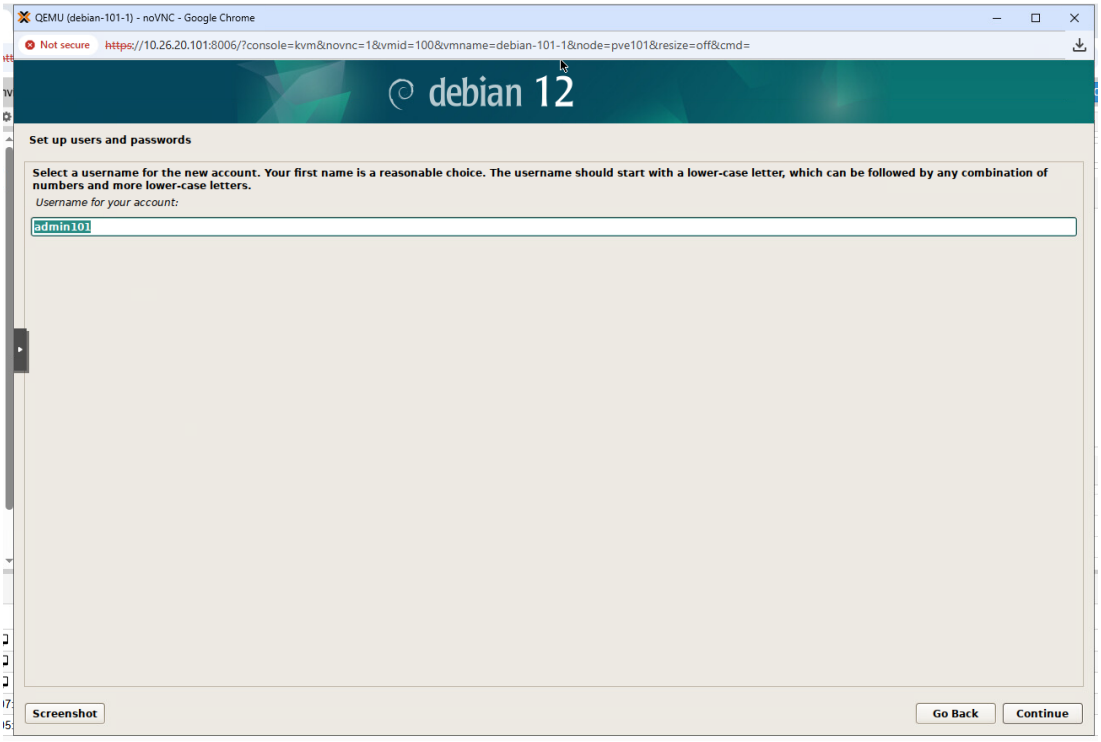
Step 8:


QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
debian 12
Set up users and passwords
You need to set a password for 'root', the system administrative account. A malicious or unqualified user with root access can have disastrous results, so you should take care to choose a root password that is not easy to guess. It should not be a word found in dictionaries, or a word that could be easily associated with you.
A good password will contain a mixture of letters, numbers and punctuation and should be changed at regular intervals.
The root user should not have an empty password. If you leave this empty, the root account will be disabled and the system's initial user account will be given the power to become root using the "sudo" command.
Note that you will not be able to see the password as you type it.
Root password:
Show Password in Clear
Please enter the same root password again to verify that you have typed it correctly.
Re-enter password to verify:
Show Password in Clear
Screenshot
Go Back
Continue

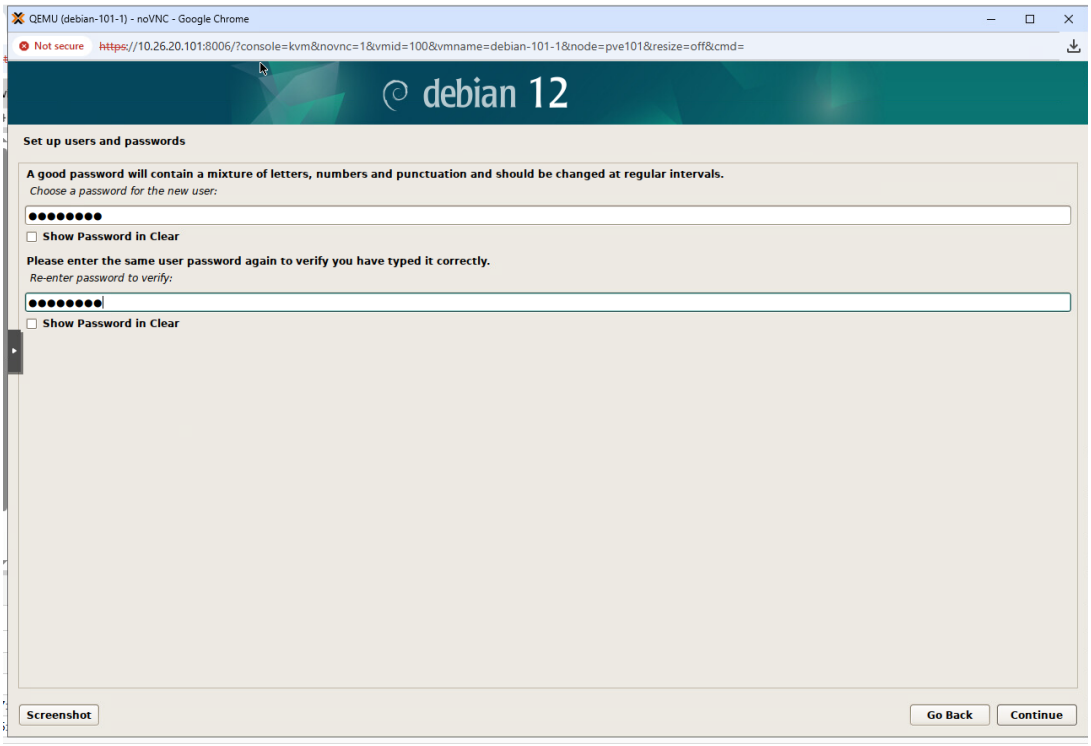
Step 9:



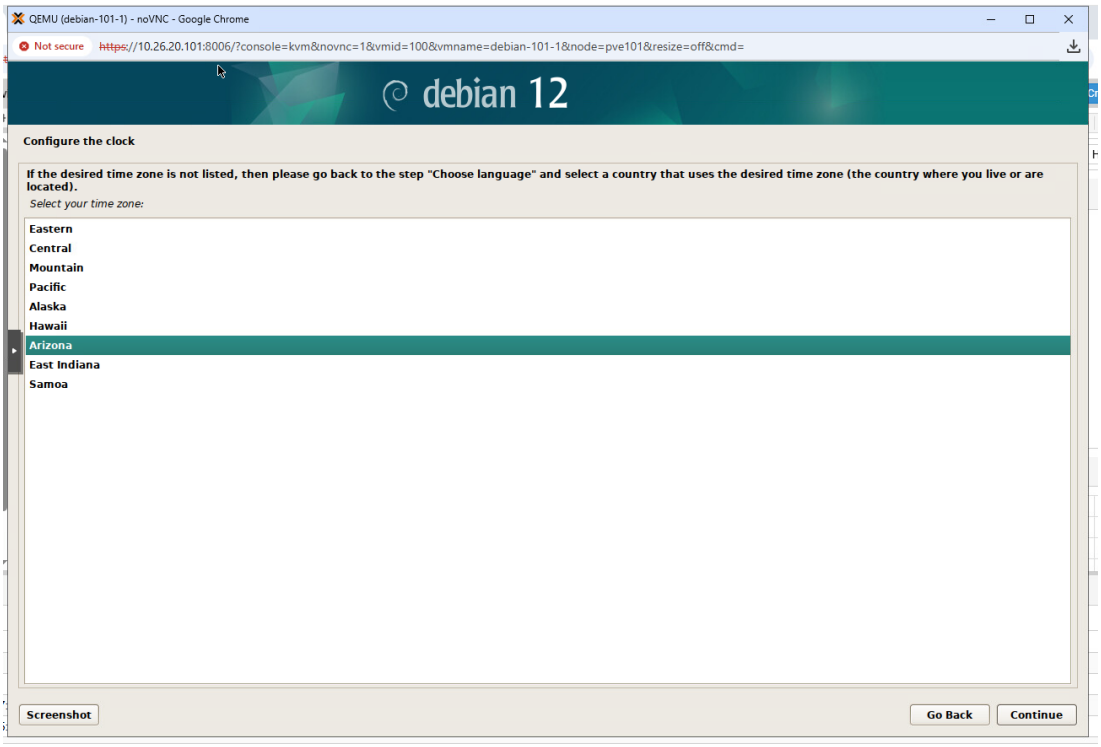
Step 10:



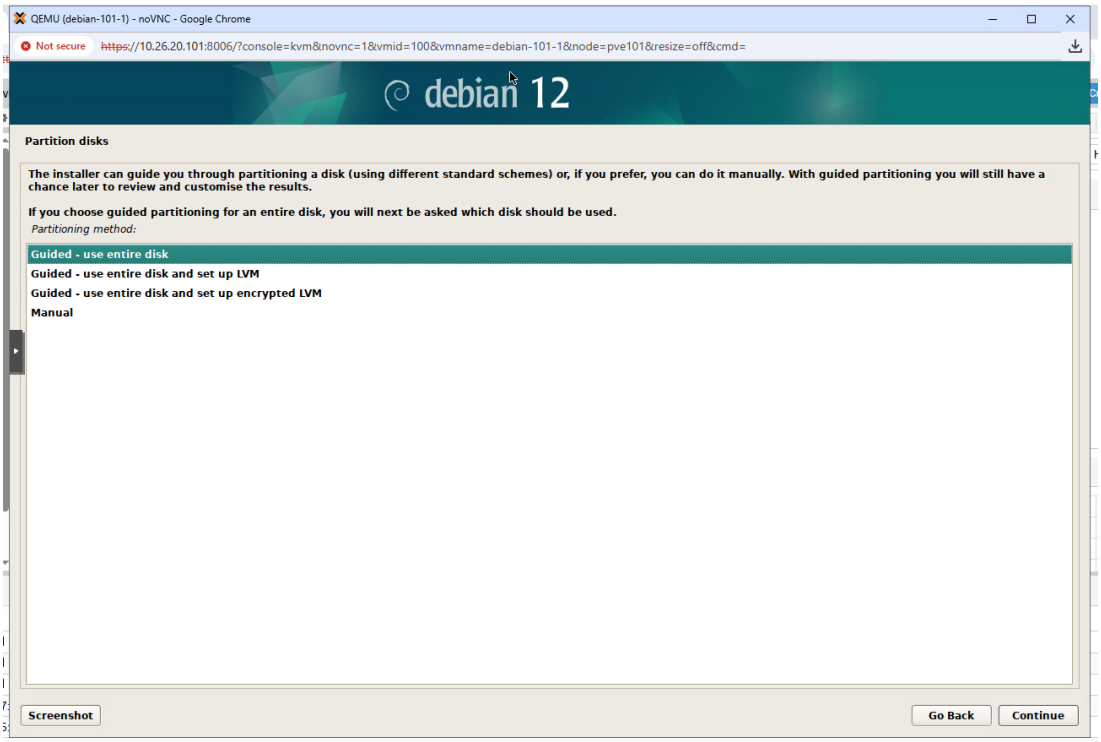
Step 11:



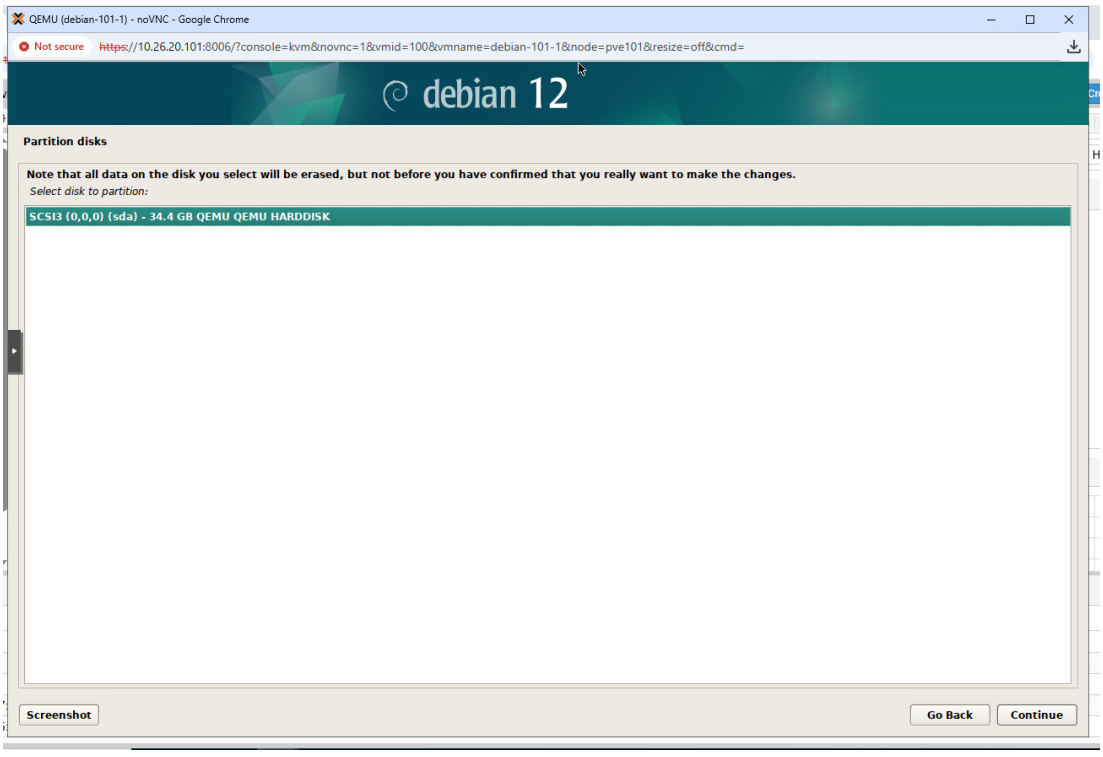
Step 12:



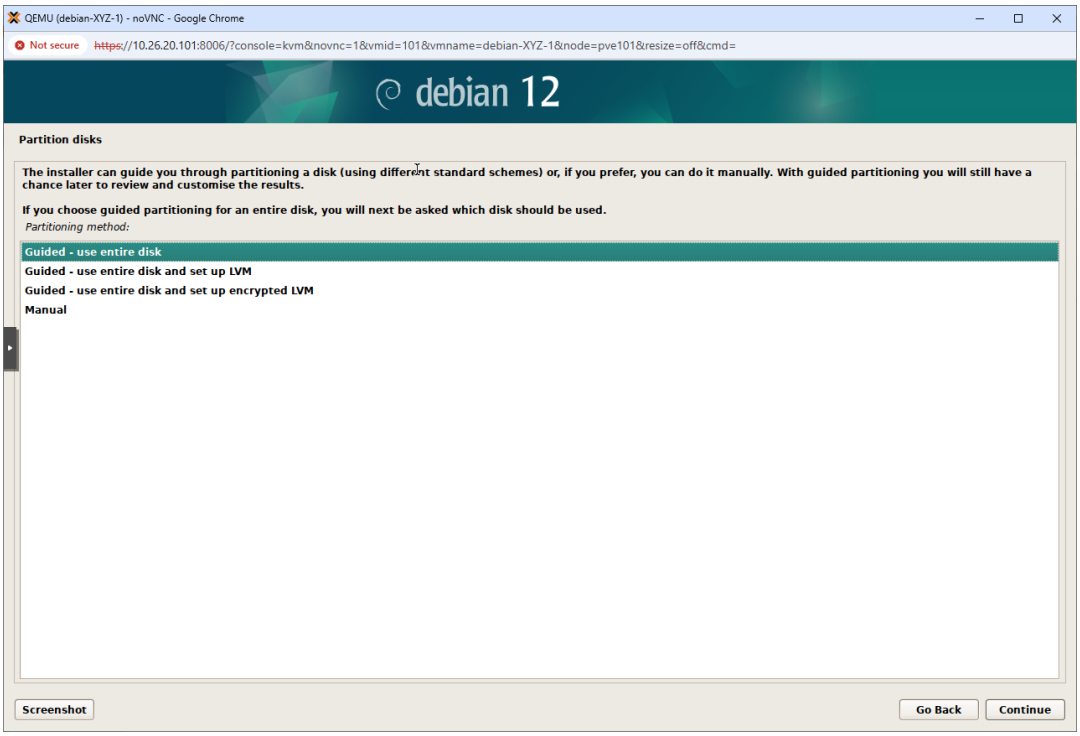
Step 13:



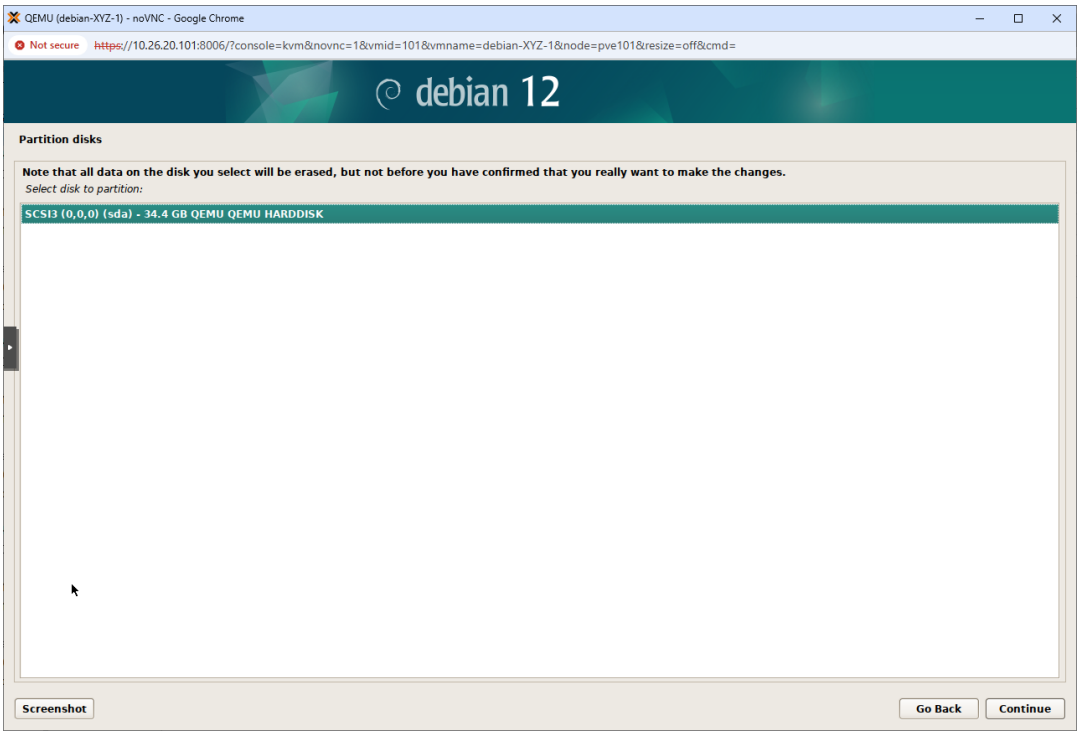
Step 14:



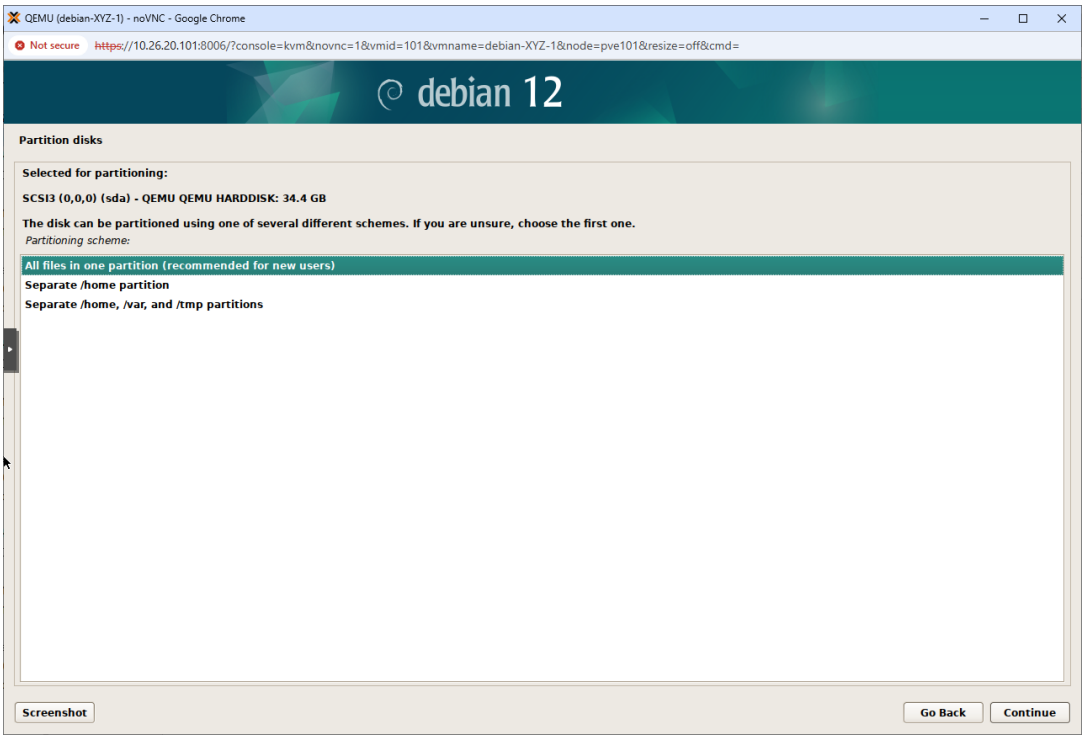
Step 15:



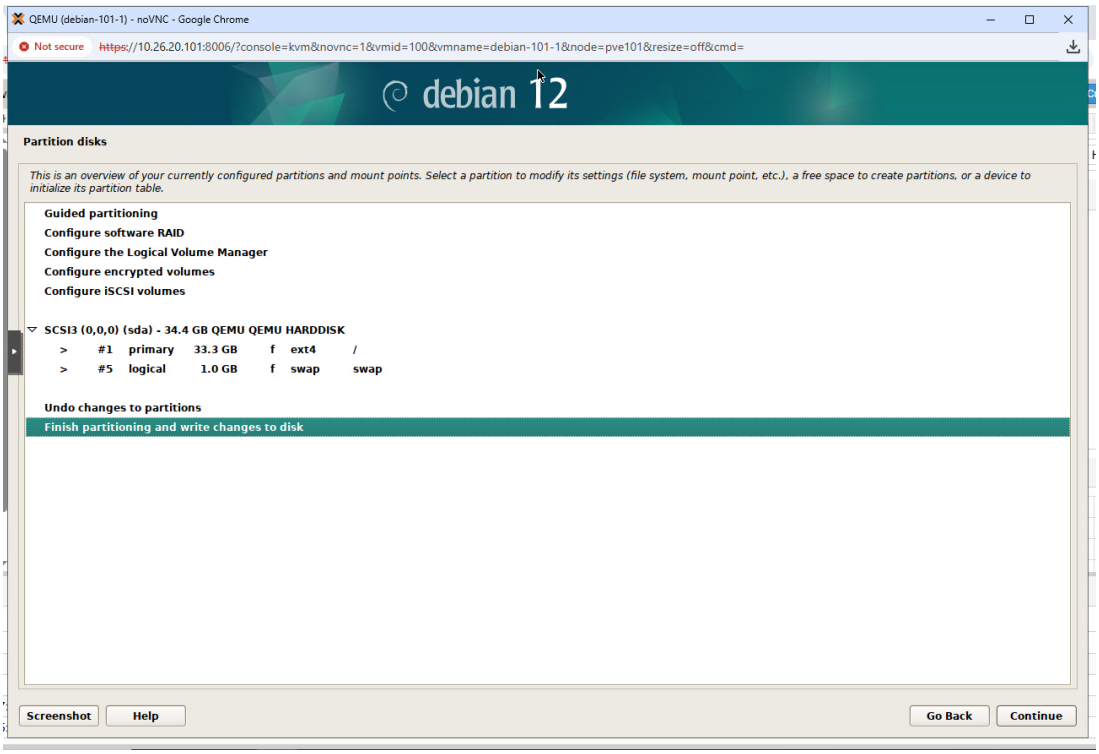
Step 16:



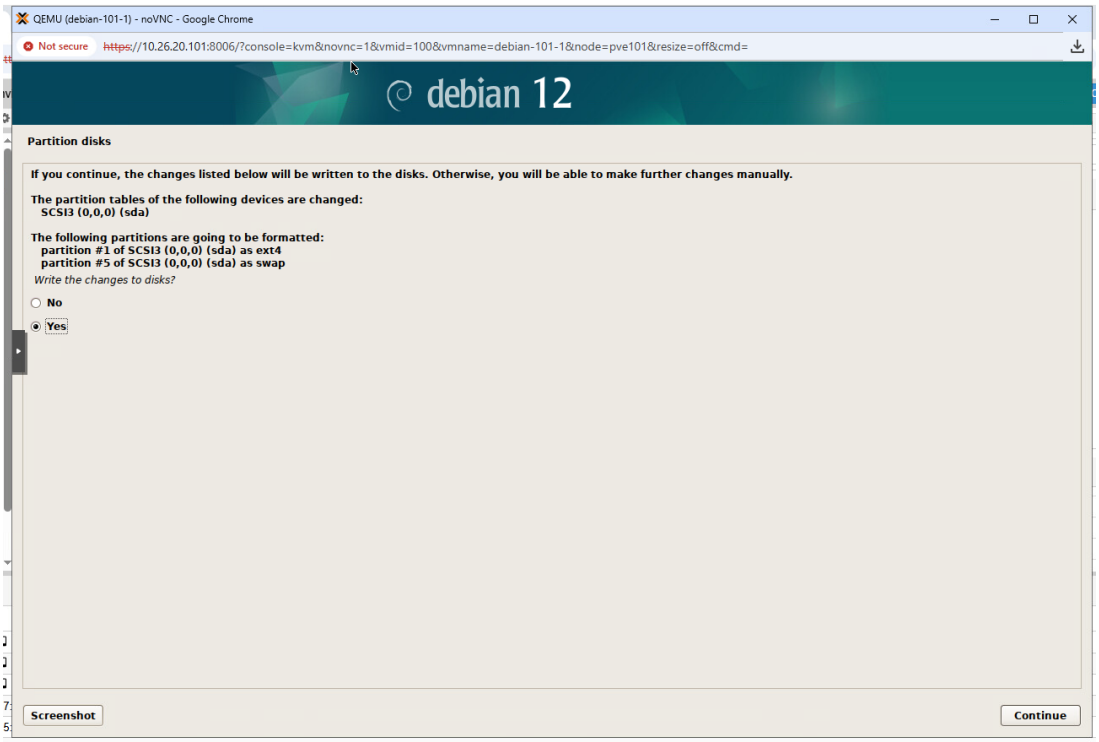
Step 17:



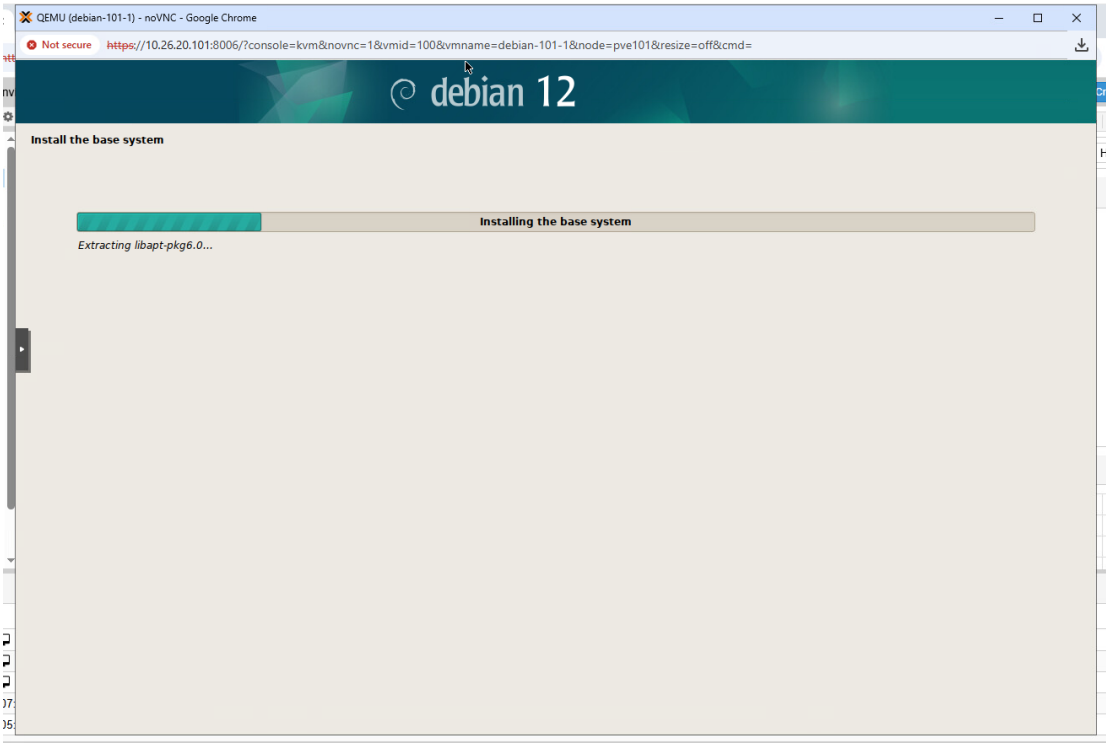
Step 18:



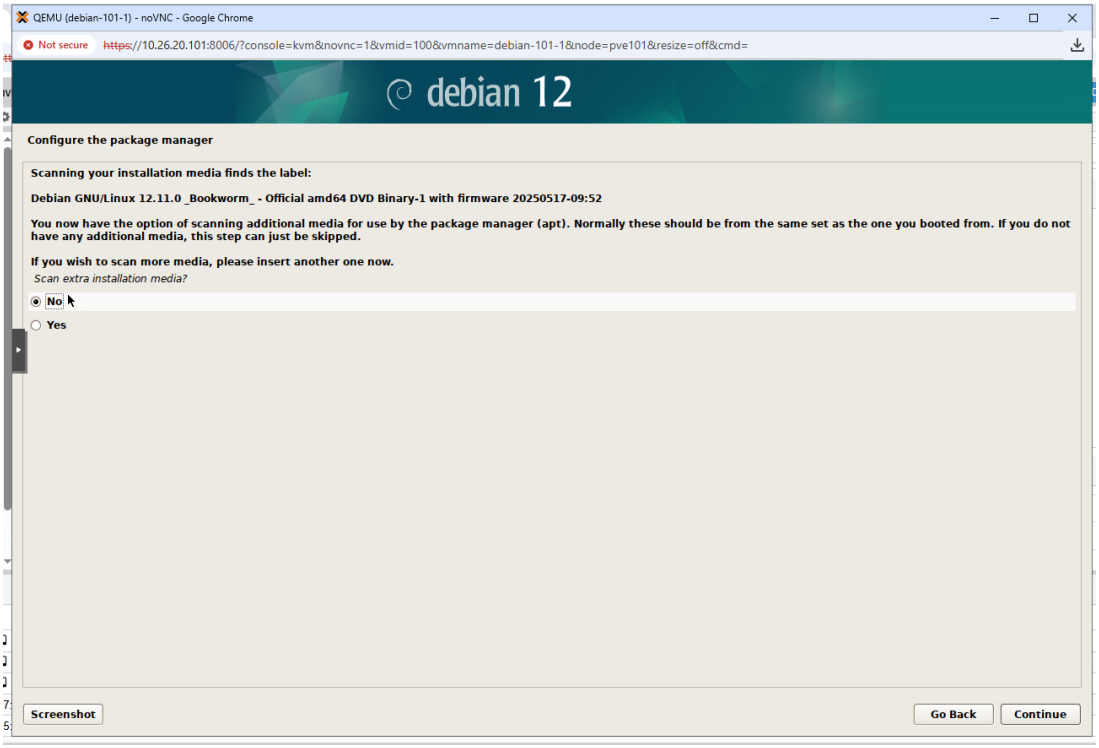
Step 19:



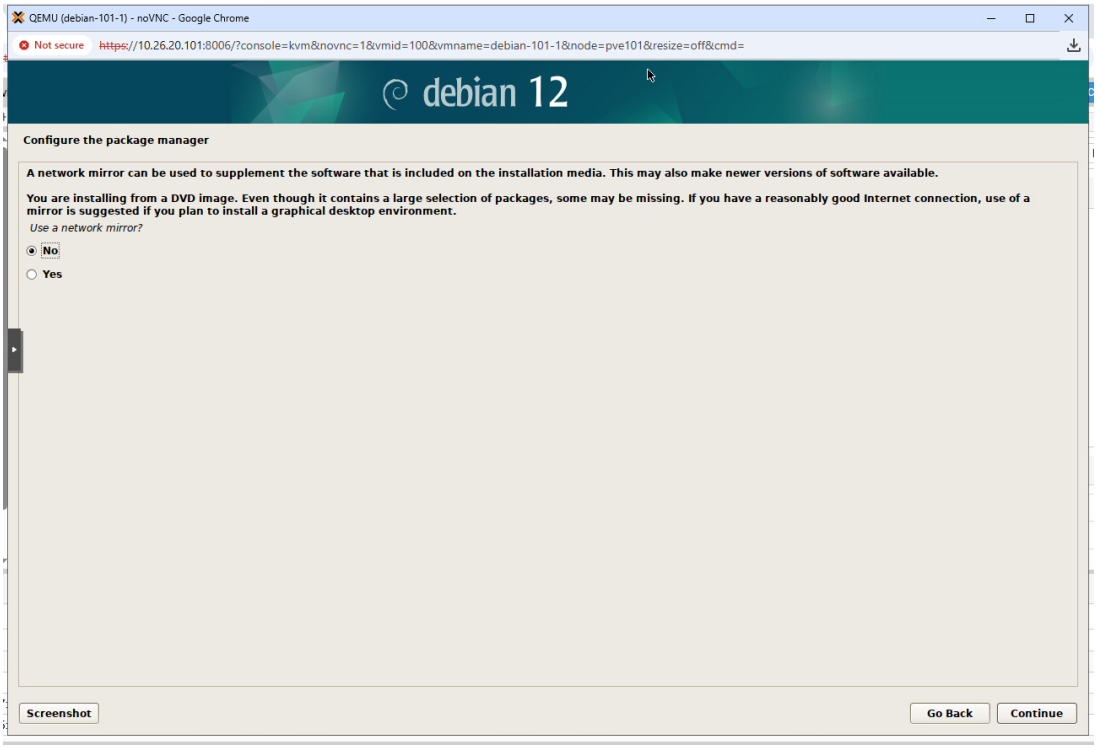
Step 20:



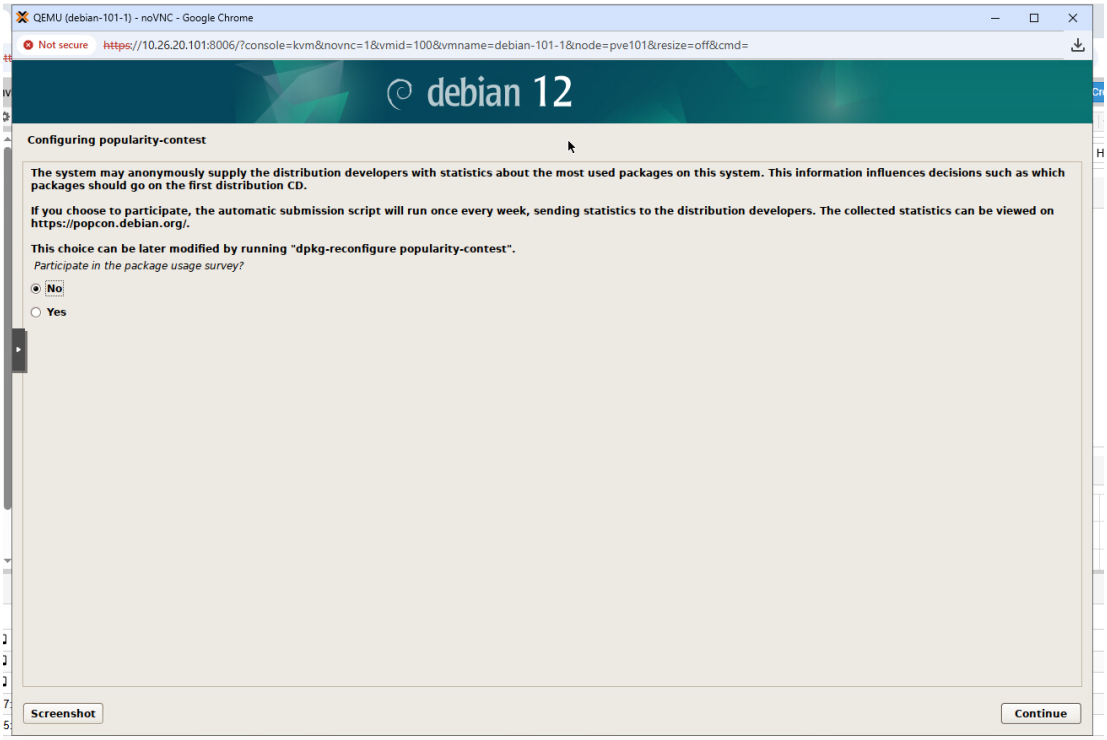
Step 21:



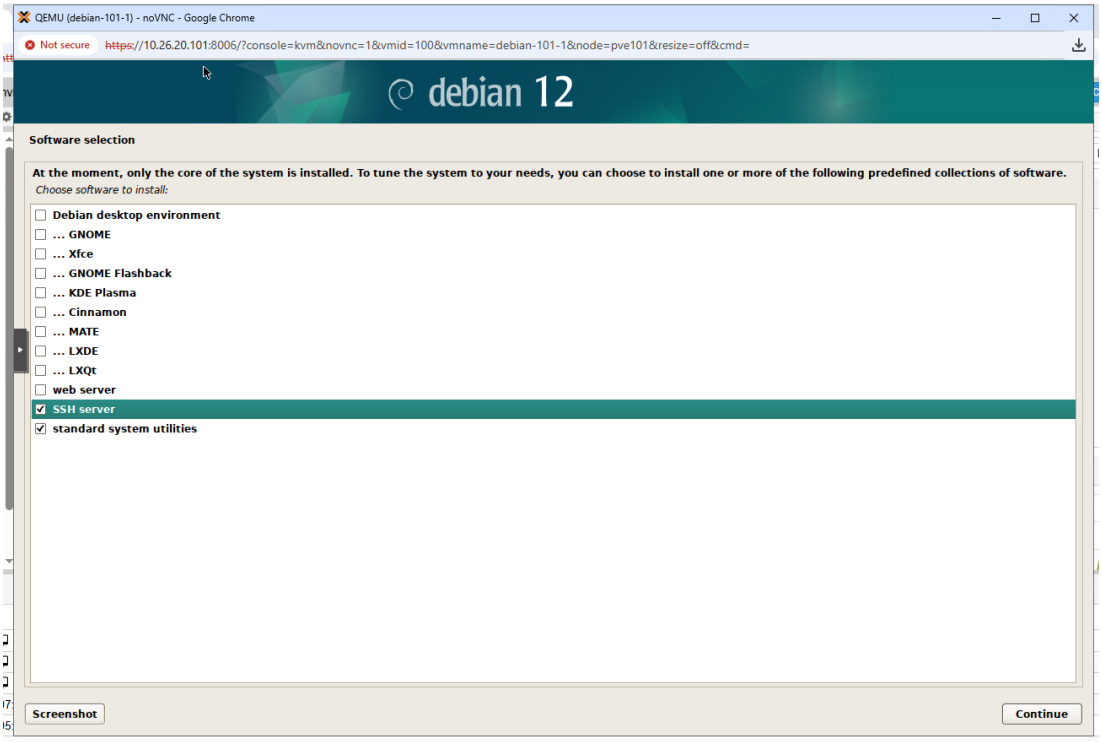
Step 22:



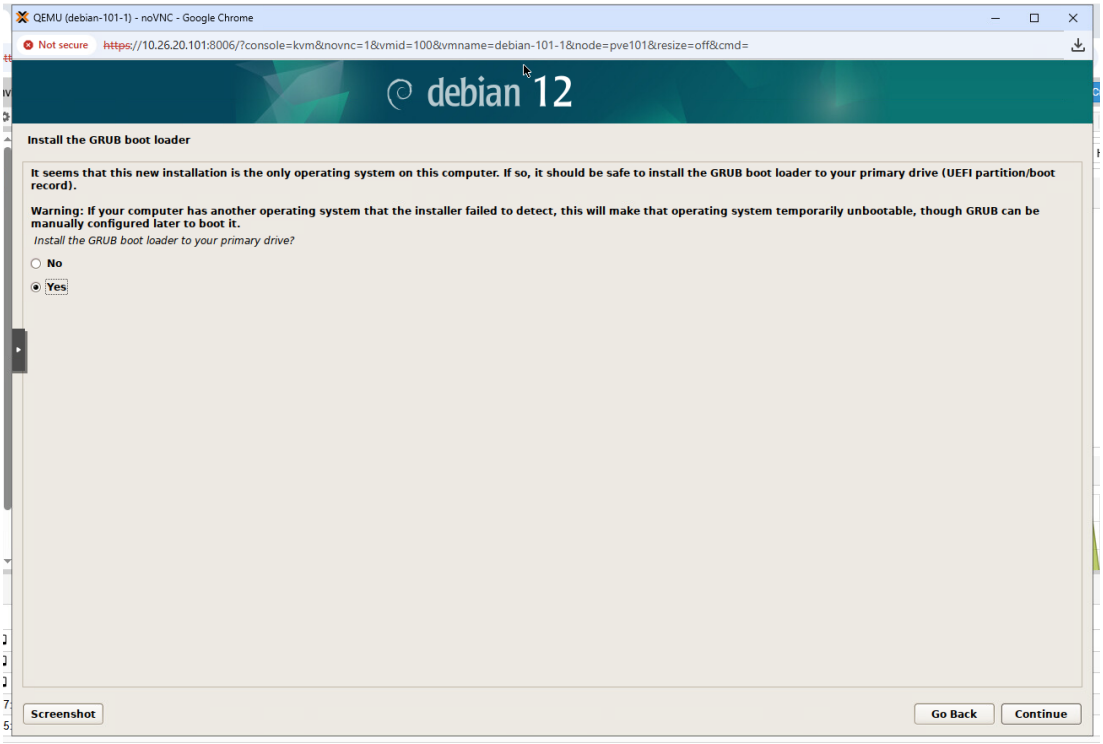
Step 23:



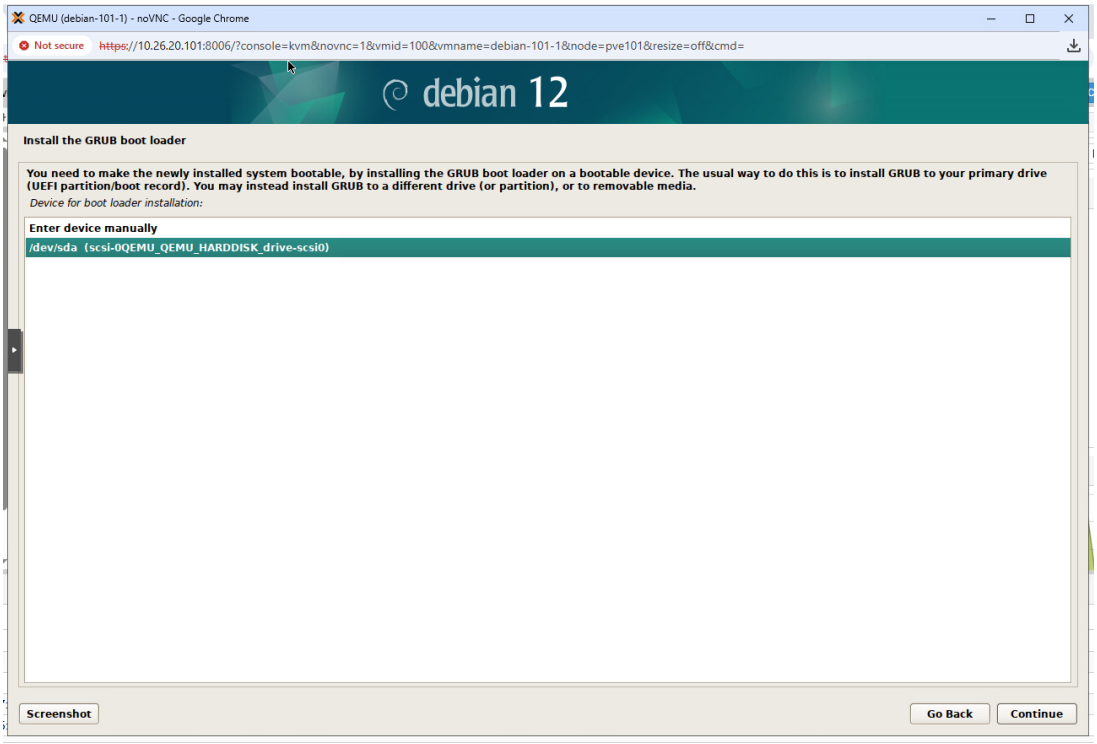
Step 24: Be sure to install SSH server (only install Debian desktop environment and GNOME if you want a Desktop GUI – longer install)



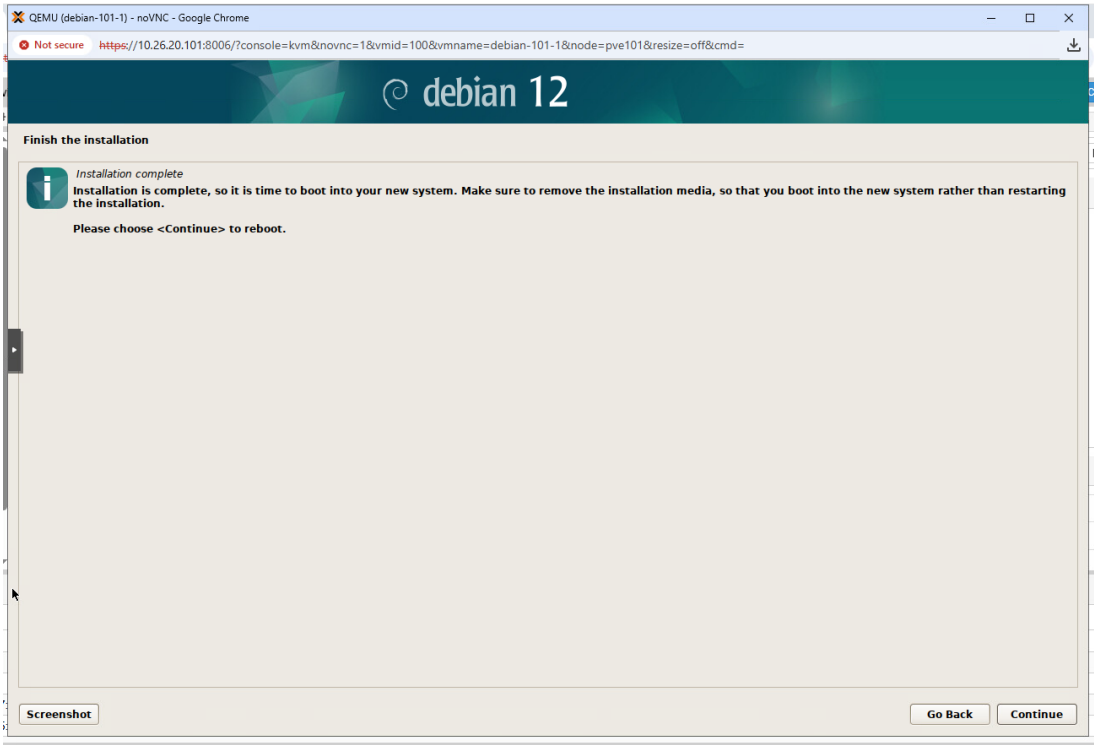
Step 25:



Step 26:



Step 27:


QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
debian 12
Finish the installation
Installation complete
Installation is complete, so it is time to boot into your new system. Make sure to remove the installation media, so that you boot into the new system rather than restarting the installation.
Please choose <Continue> to reboot.
Screenshot
Go Back
Continue

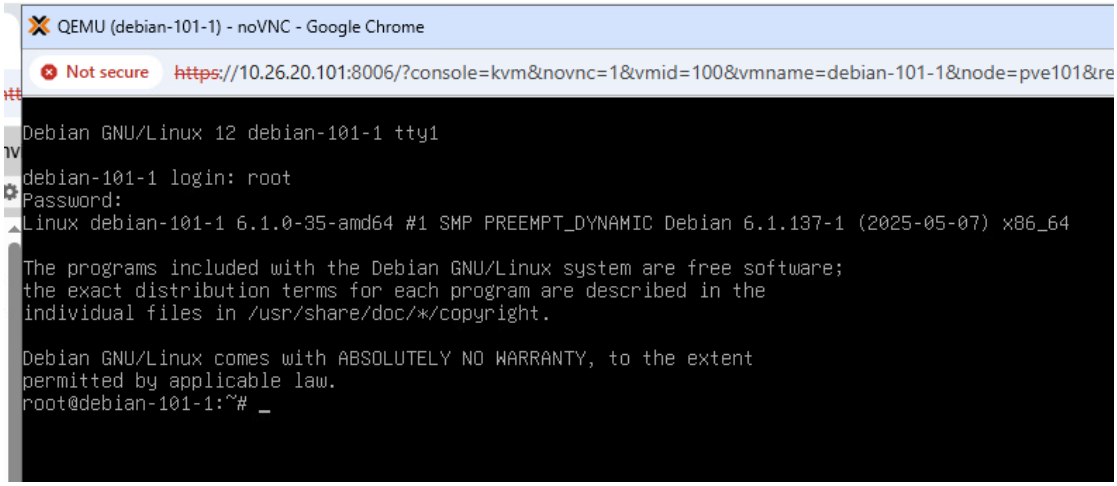
SBS LAB (CLI) – Install open-vm-tools

Long ago, Linux distributions took over the creation and maintenance of distro-specific drivers and utilities from hypervisor vendors (like Proxmox and VMware). This comes in the form of open-vm-tools. While Microsoft remains steadfast in NOT providing such a package, leaving us to use VirtIO for windows, open-vm-tools is the choice for Linux.

1. Now we need to install open-vm-tools

   Step 1: Log on as: root

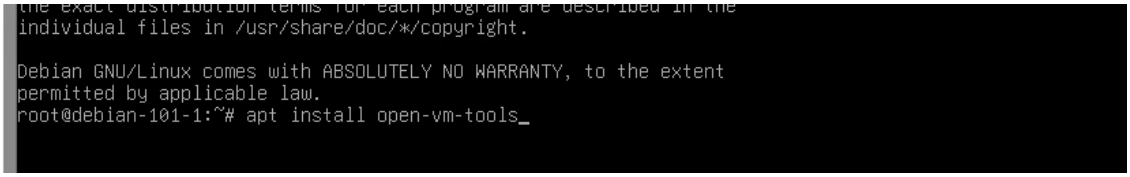
   e.g.:



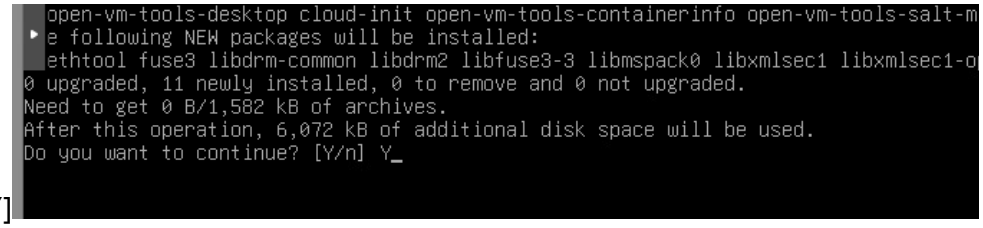
   NOTE : root login is permitted only from the console unless you change sshd_config to allow root login

   Command #1 run: Apt install open-vm-tools

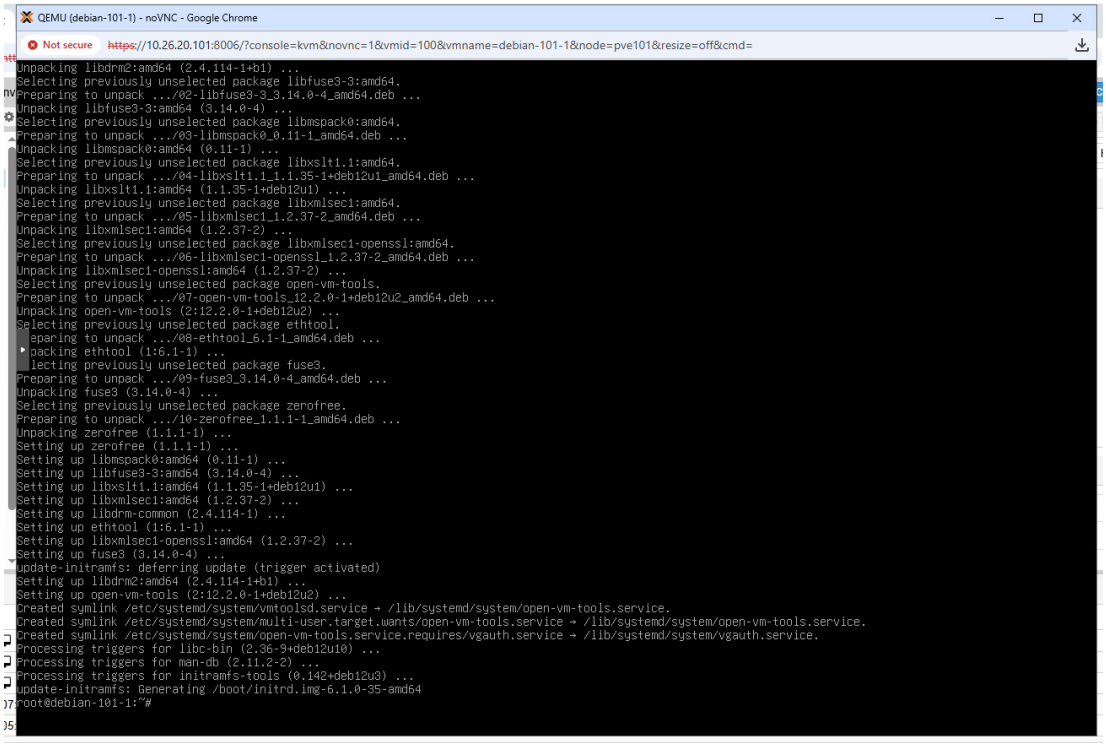
   e.g.:



   Step 2: [Y]



e.g.:

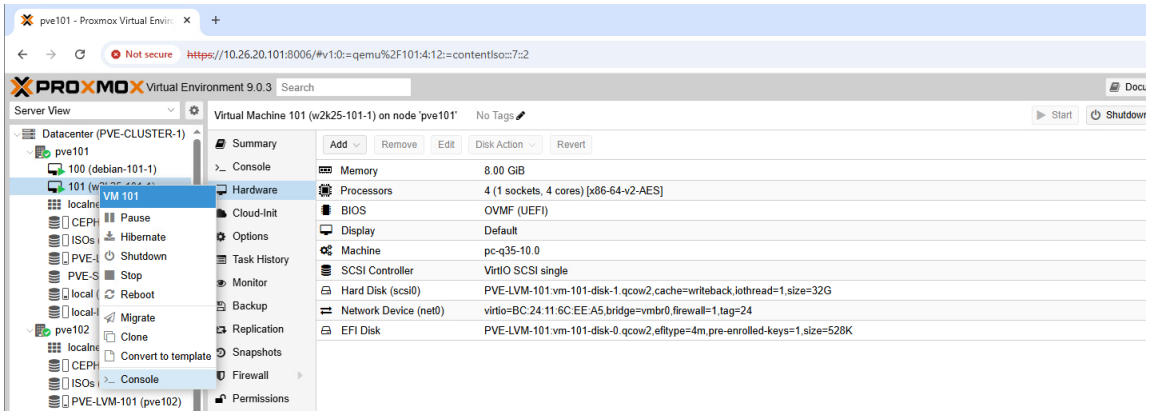

QEMU (debian-101-1) - noVNC - Google Chrome
Not secure
https://10.26.20.101:8006/?console=kvm&novnc=1&vmid=100&vmname=debian-101-1&node=pve101&resize=off&cmd=
Unpacking libdrm2:amd64 (2.4.114-1+b1) ...
Selecting previously unselected package libfuse3-3:amd64.
Preparing to unpack .../02-libfuse3-3_3.14.0-4_amd64.deb ...
Unpacking libfuse3-3:amd64 (3.14.0-4) ...
Selecting previously unselected package libmspack0:amd64.
Preparing to unpack .../03-libmspack0_0.11-1_amd64.deb ...
Unpacking libmspack0:amd64 (0.11-1) ...
Selecting previously unselected package libxslt1.1:amd64.
Preparing to unpack .../04-libxslt1.1_1.1.35-1+deb12u1_amd64.deb ...
Unpacking libxslt1.1:amd64 (1.1.35-1+deb12u1) ...
Selecting previously unselected package libxmlsec1:amd64.
Preparing to unpack .../05-libxmlsec1_1.2.37-2_amd64.deb ...
Unpacking libxmlsec1:amd64 (1.2.37-2) ...
Selecting previously unselected package libxmlsec1-openssl:amd64.
Preparing to unpack .../06-libxmlsec1-openssl_1.2.37-2_amd64.deb ...
Unpacking libxmlsec1-openssl:amd64 (1.2.37-2) ...
Selecting previously unselected package open-vm-tools.
Preparing to unpack .../07-open-vm-tools_12.2.0-1+deb12u2_amd64.deb ...
Unpacking open-vm-tools (2:12.2.0-1+deb12u2) ...
Selecting previously unselected package ethtool.
eparing to unpack .../08-ethtool_6.1-1_amd64.deb ...
packing ethtool (1:6.1-1) ...
lecting previously unselected package fuse3.
Preparing to unpack .../09-fuse3_3.14.0-4_amd64.deb ...
Unpacking fuse3 (3.14.0-4) ...
Selecting previously unselected package zerofree.
Preparing to unpack .../10-zerofree_1.1.1-1_amd64.deb ...
Unpacking zerofree (1.1.1-1) ...
Setting up zerofree (1.1.1-1) ...
Setting up libmspack0:amd64 (0.11-1) ...
Setting up libfuse3-3:amd64 (3.14.0-4) ...
Setting up libxslt1.1:amd64 (1.1.35-1+deb12u1) ...
Setting up libxmlsec1:amd64 (1.2.37-2) ...
Setting up libdrm-common (2.4.114-1) ...
Setting up ethtool (1:6.1-1) ...
Setting up libxmlsec1-openssl:amd64 (1.2.37-2) ...
Setting up fuse3 (3.14.0-4) ...
update-initramfs: deferring update (trigger activated)
Setting up libdrm2:amd64 (2.4.114-1+b1) ...
Setting up open-vm-tools (2:12.2.0-1+deb12u2) ...
Created symlink /etc/systemd/system/vmtoolsd.service → /lib/systemd/system/open-vm-tools.service.
Created symlink /etc/systemd/system/multi-user.target.wants/open-vm-tools.service → /lib/systemd/system/open-vm-tools.service.
Created symlink /etc/systemd/system/open-vm-tools.service.requires/vgauth.service → /lib/systemd/system/vgauth.service.
Processing triggers for libc-bin (2.36-9+deb12u10) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for initramfs-tools (0.142+deb12u3) ...
update-initramfs: Generating /boot/initrd.img-6.1.0-35-amd64
root@debian-101-1:~#

Managing VMs on PVE

SBS LAB – (GUI) Live migrating VMs between PVE nodes

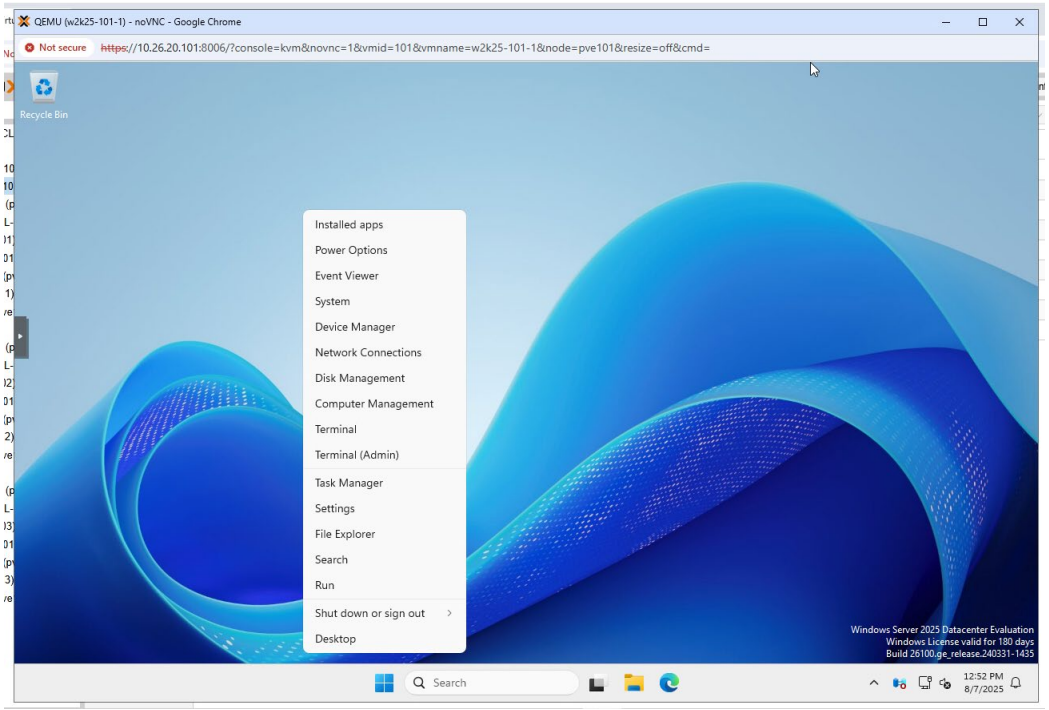
Step 1: Open a console to a running VM

e.g.:



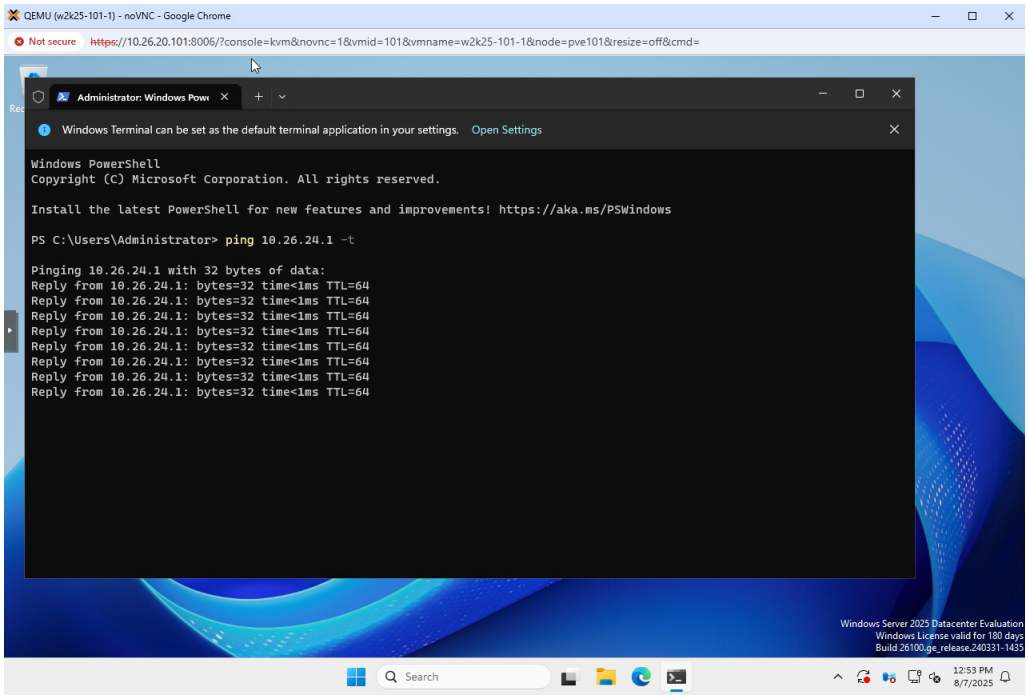
Step 2: Start a Terminal (admin)

e.g.:



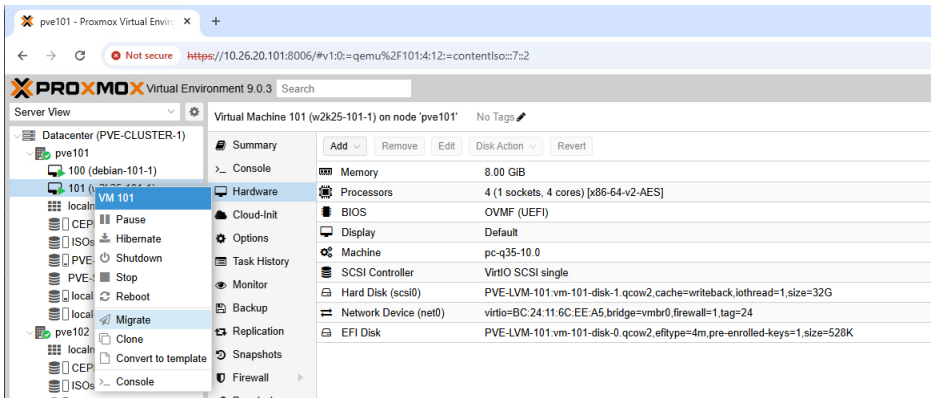
Step 3: Ping the gateway (or any other pingable IP) continuously

Step 4:



Step 5: Select your w2k25-XYZ-1 VM > Right-click > Migrate

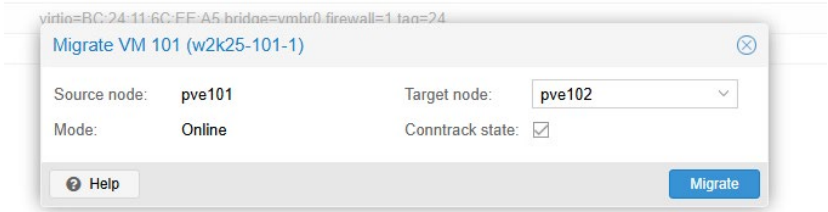
e.g.:

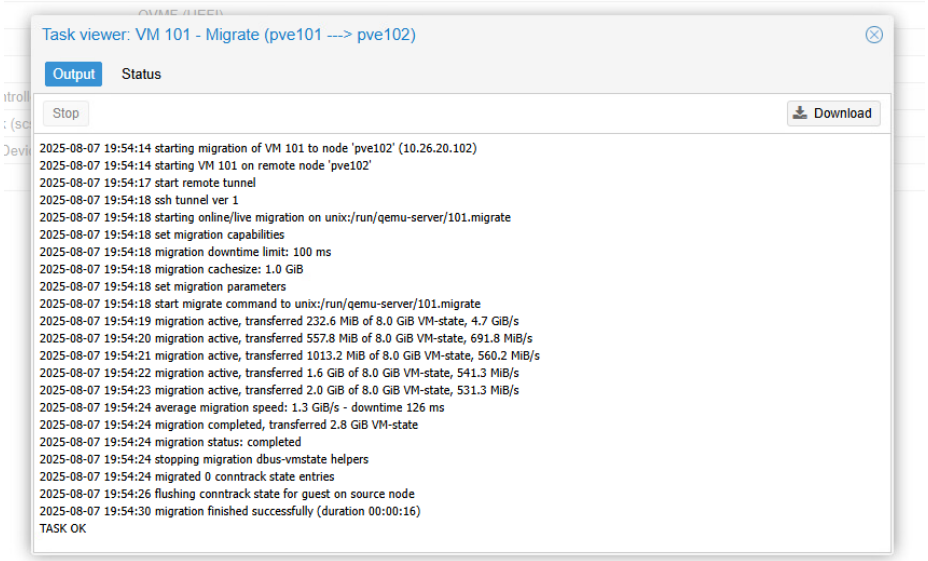


Step 6: Select destination node (any node)

Step 7: Migrate

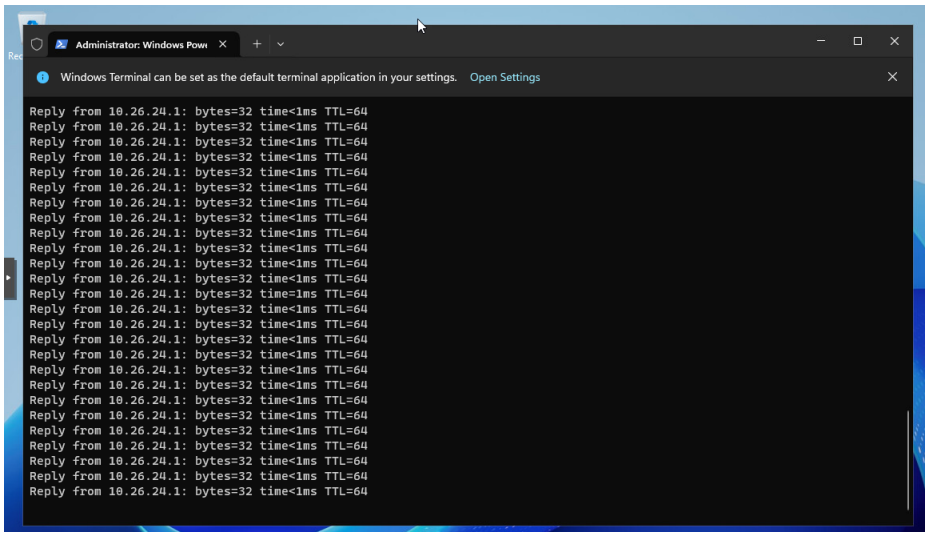
e.g.:





e.g.:

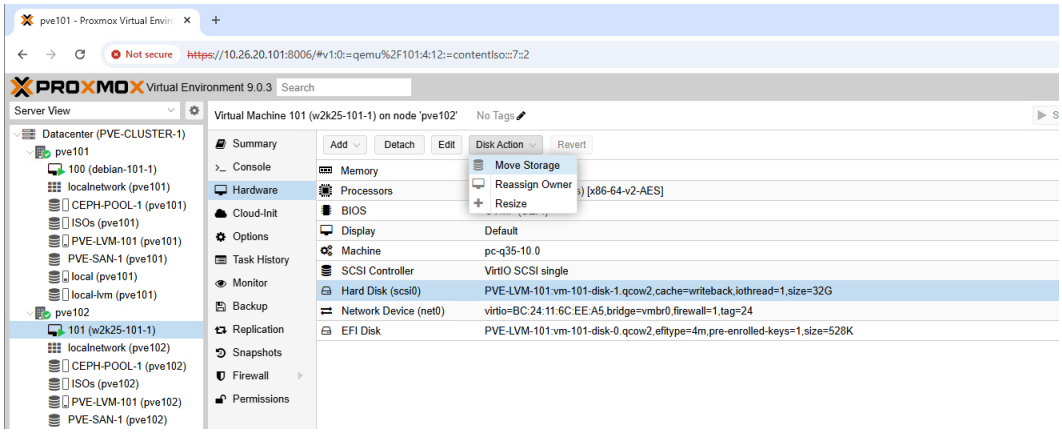
Step 8: Observe ping

e.g.: 

SBS LAB –(GUI) Live storage migration for a VM

Step 1: Select your w2k25-XYZ-1 VM > Hardware > Hard Disk (s) > Disk Action > Move Storage

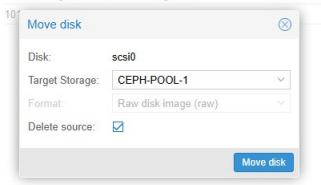
e.g.:



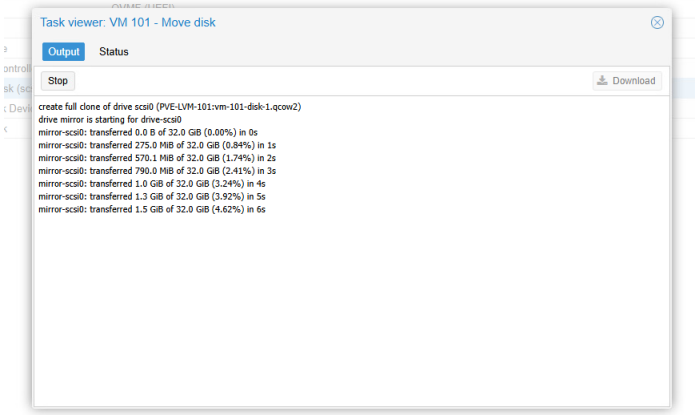
Step 2: Target Storage: CEPJ-POOL-1

Step 3: Delete source: checked (this is important or it will leave disk copies behind)

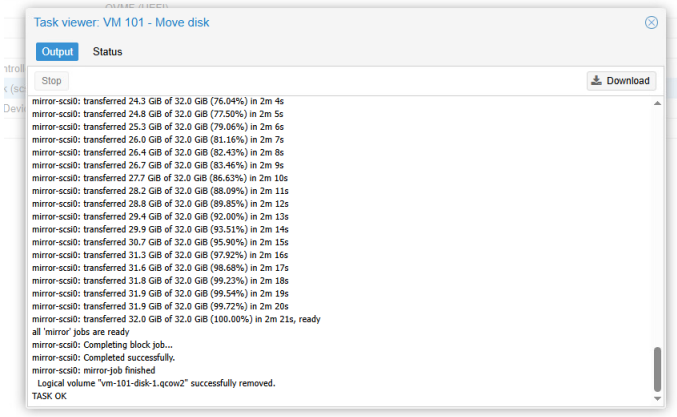
e.g.:



e.g.:



e.g.:



e.g.:

SBS LAB –(GUI) VM Snapshots on PVE

Snapshots are a particularly useful feature of any Hypervisor, but you have to be careful. Snapshots represent a delta from the point where you created the snapshot, and that delta will continue to grow as long as it exists, potentially consuming all available storage!

Snapshots are part of a process, not a permanent form of backup.

Here's a scenario which is appropriate for Snapshot use:

> Administrator is about to update certificates on a complex application spanning multiple VMs. A snapshot is made prior to updating certificates. The functionality of the certificates is validated. Snapshots are removed.

Here's a scenario where snapshots are NOT appropriate:

> Administrator has finished configuring a complex server/app and creates a Snapshot to protect their work. NOTE: A one-time backup should be performed in this situation

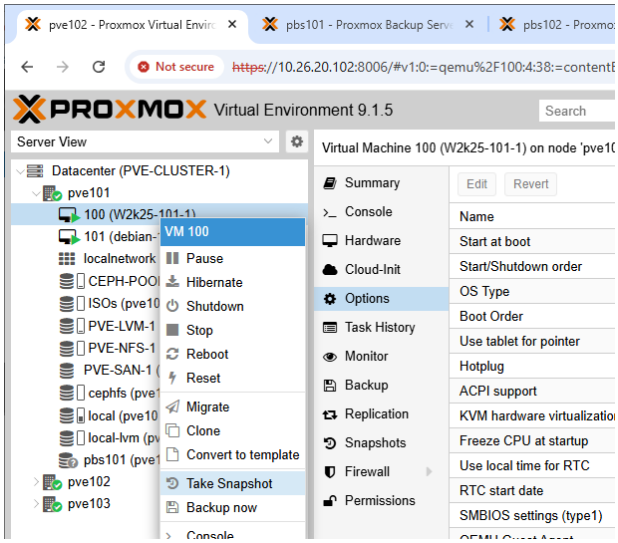
Snapshots come in two forms: With VM memory (Include RAM) and without VM memory.

When you include RAM, the snapshot takes longer to create (the VM is still functional during this time) but preserves the exact state of the VM at the time the snapshot was created. Include RAM snapshots are the best choice when used as part of a procedural administrative process like upgrades and certificates.

When you don't include RAM, the snapshot is almost instantaneous, but rolling back to any snapshot point involves booting the VM from power off. This is the type of snapshot used by backup applications to quiesce the VM and capture transactional data accurately. Backup snapshots are removed automatically when the backup is complete.

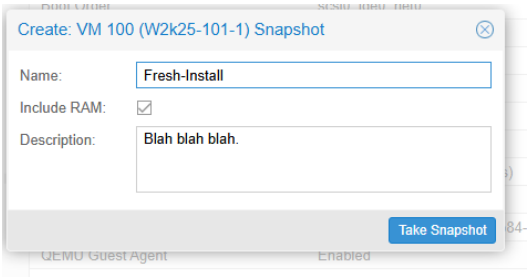
Step 1: Choose your w2k25-XYZ-1 VM > Right-click > Take Snapshot

e.g.:



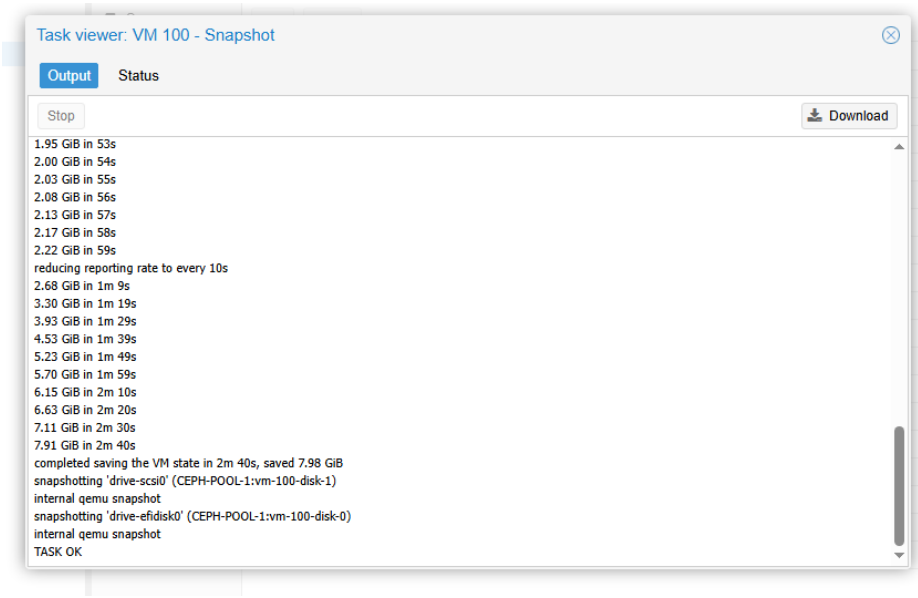
Step 2: Name the snapshot (no spaces)

e.g.:



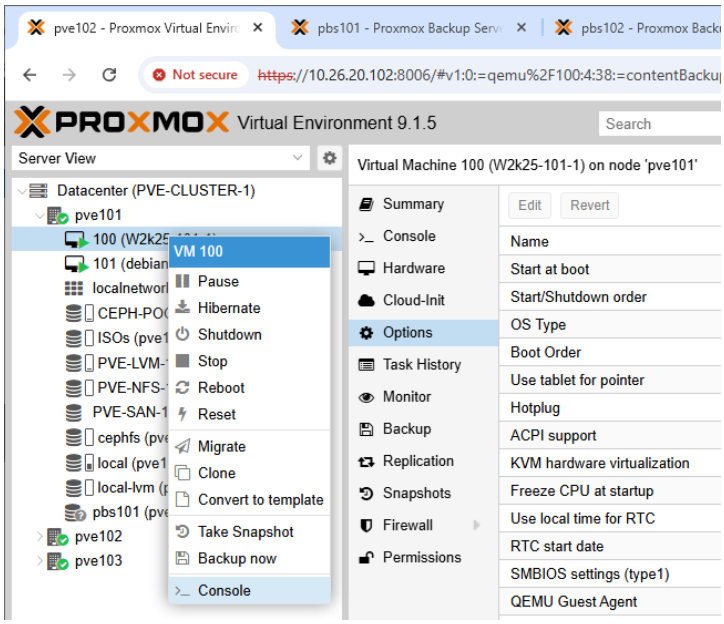
Step 3: Wait for completion

e.g.:

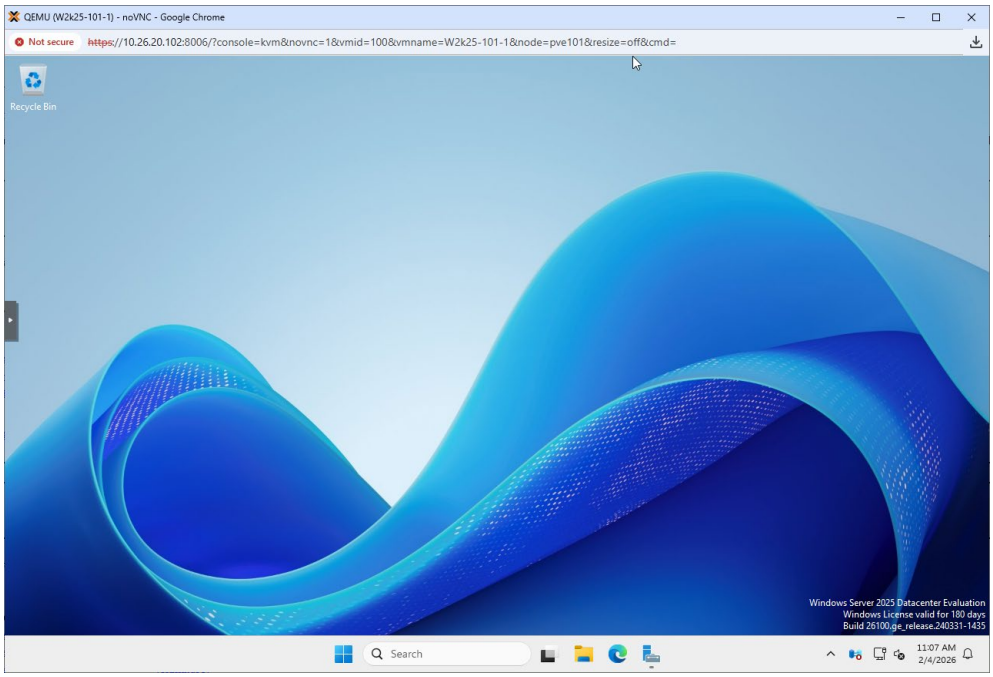


Step 4: Choose your w2k25-XYZ-1 VM > Right-click > Console

e.g.:

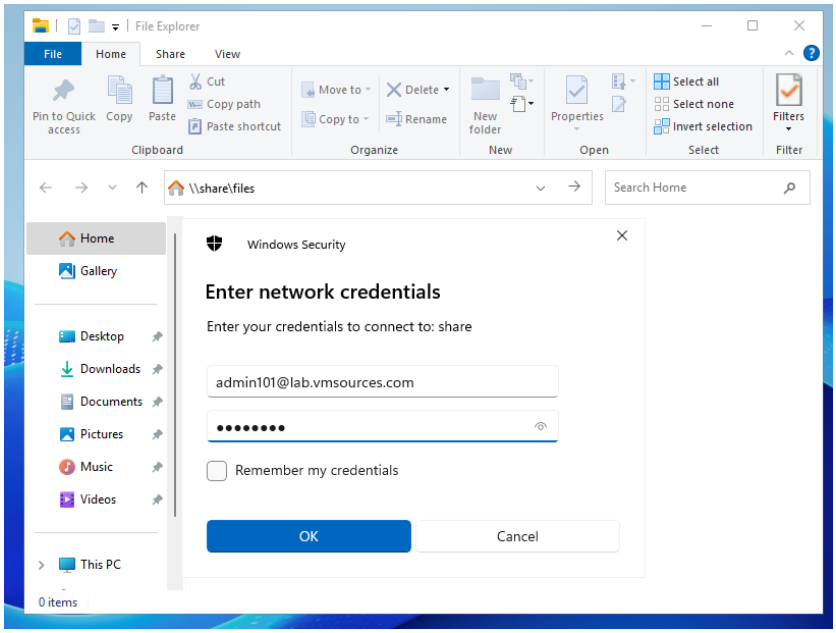


e.g.:



Step 5: Browse to: \\share\files

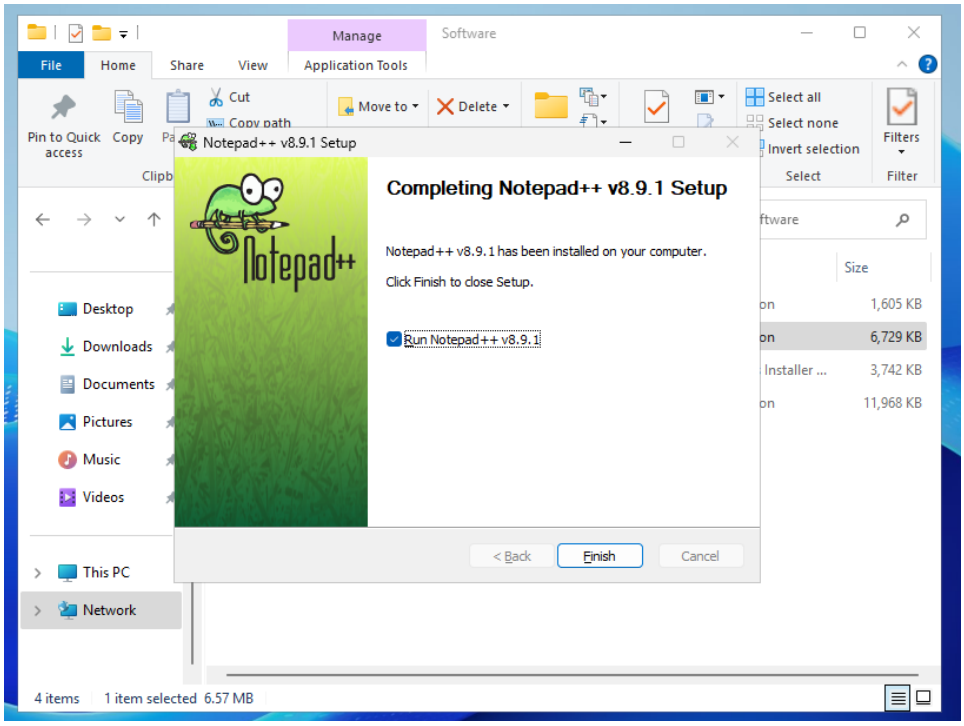
e.g.:



Step 6: Install Notepad++ (or any other packages available in \\share\files\Software)

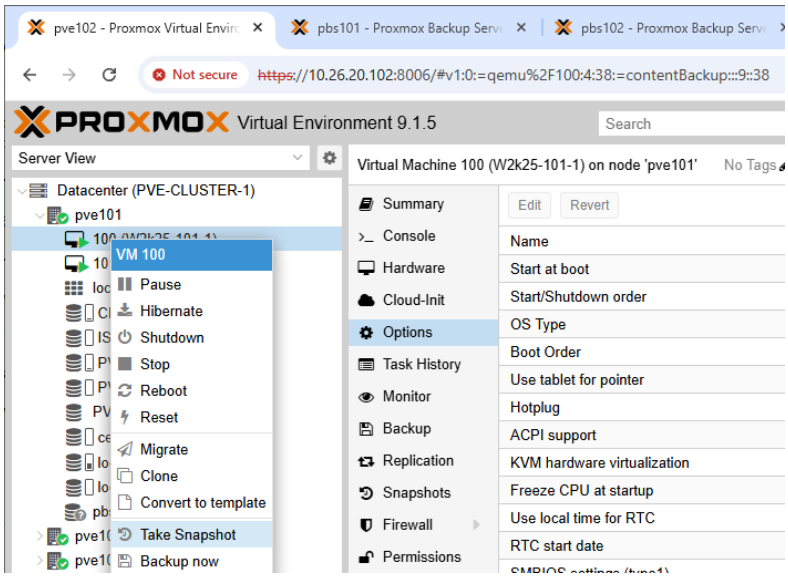
NOTE : Accept all defaults during the installer

e.g.:

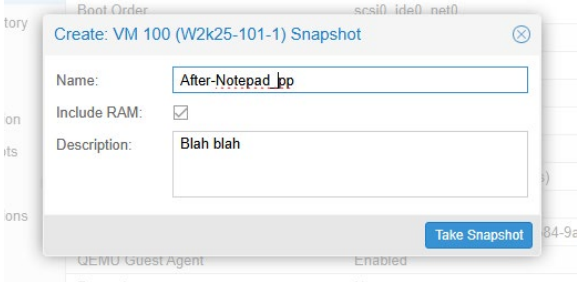


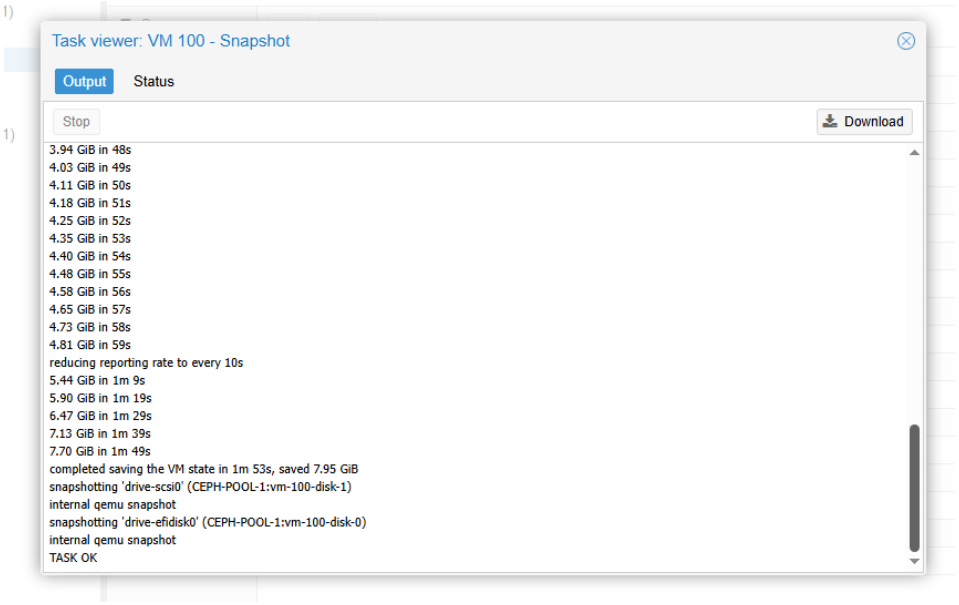
Step 7: Take another snapshot of w2k25-XYZ-1

e.g.:



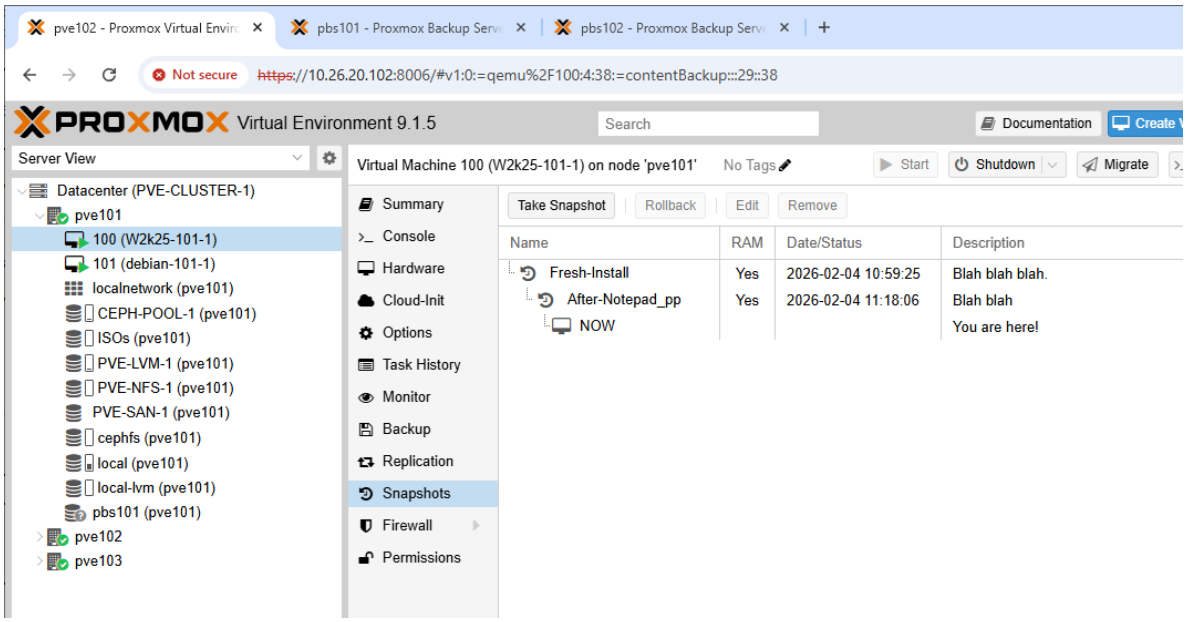
e.g.:





e.g.:

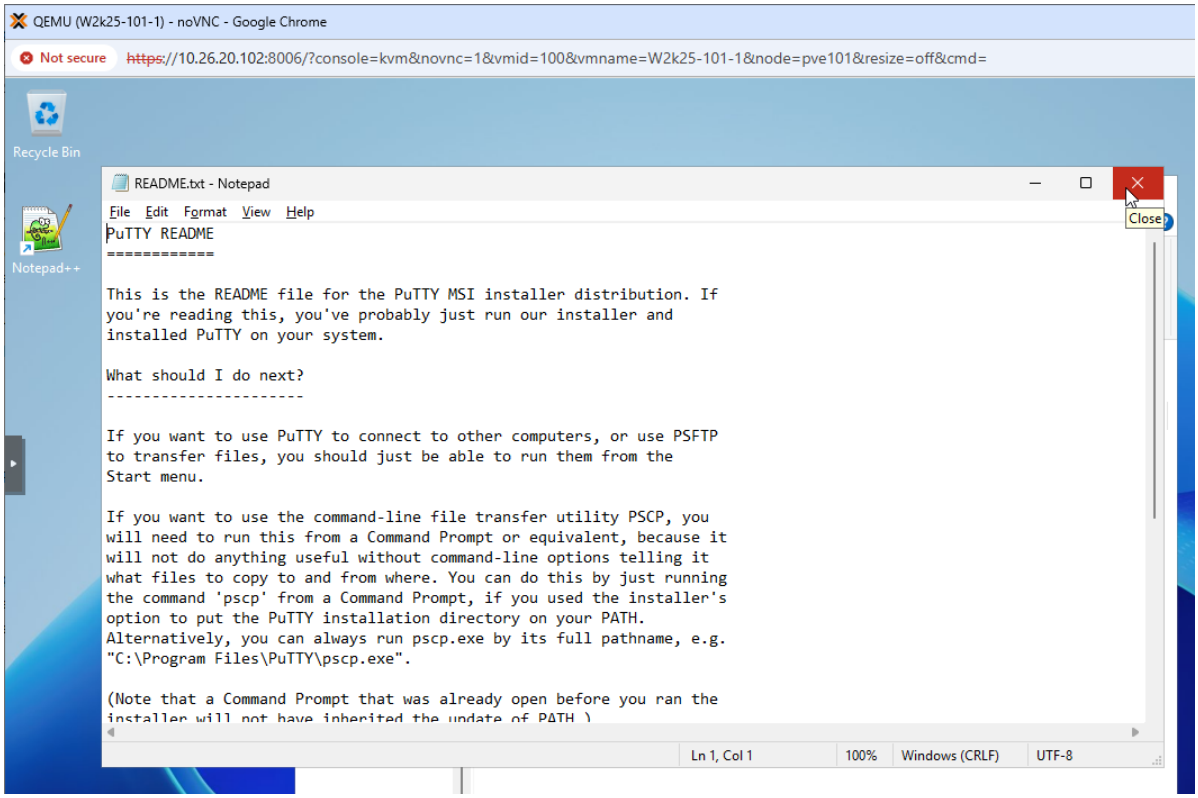
Step 8: View Snapshot Hierarchy: pveXYZ > w2k25-XYZ-1 > Snapshots



e.g.:

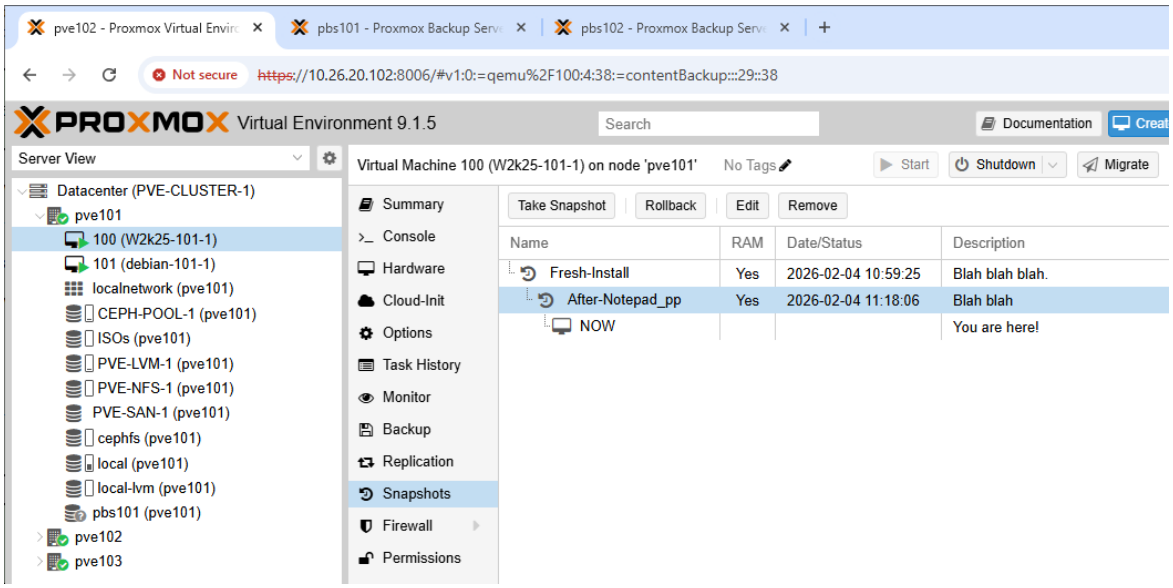
Step 9: Now make an additional change to the VM (like installing another app)

e.g.:

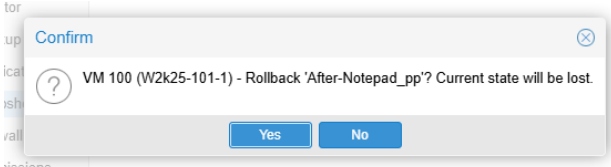


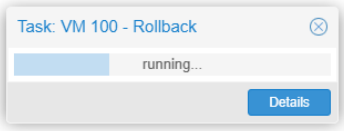
Step 10: Go back to: pveXYZ > w2k25-XYZ-1 > Snapshots and select: Rollback

e.g.:



e.g.:

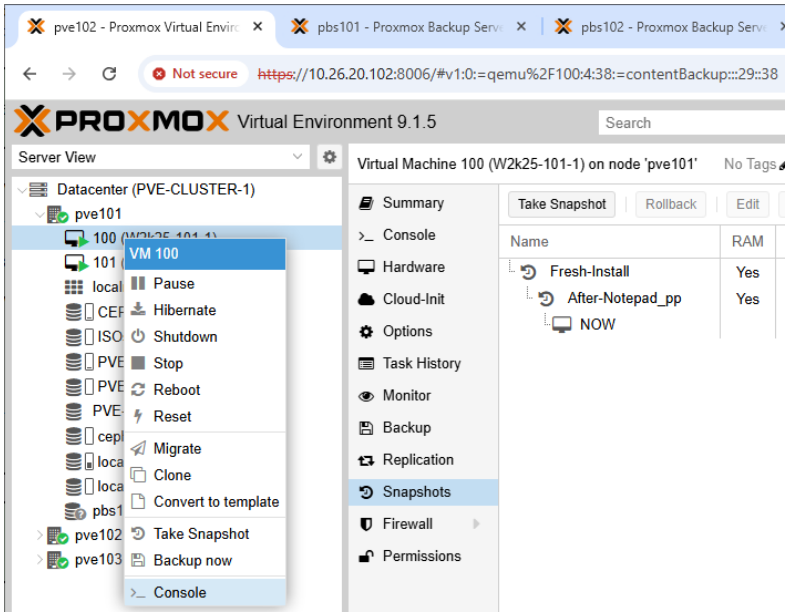




e.g.:

Step 11: You will need to open a fresh console to your VM after rollback

e.g.:

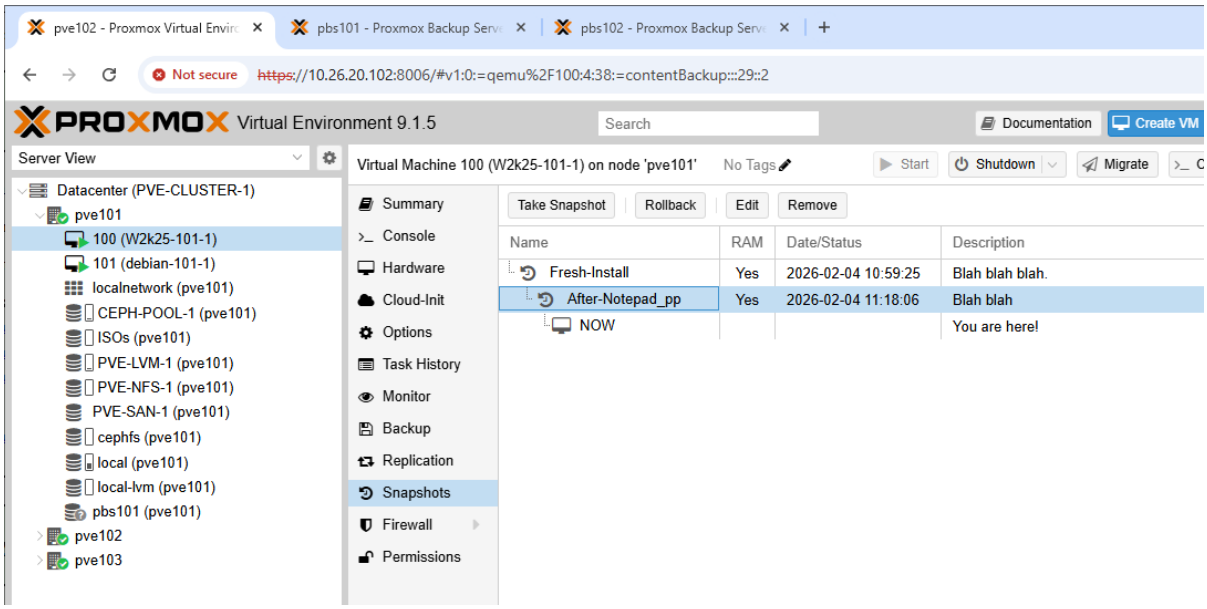


2. Your second installation is gone, but the first remains

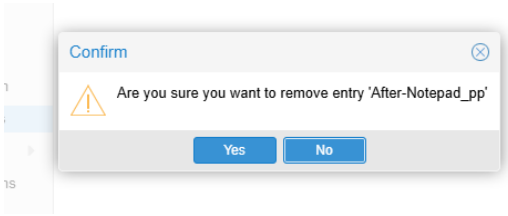
Step 1: Choose the state you want your VM to be and select: Rollback (unless present state is acceptable.

Step 2: Remove all snapshots

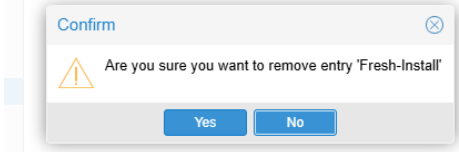
e.g.:



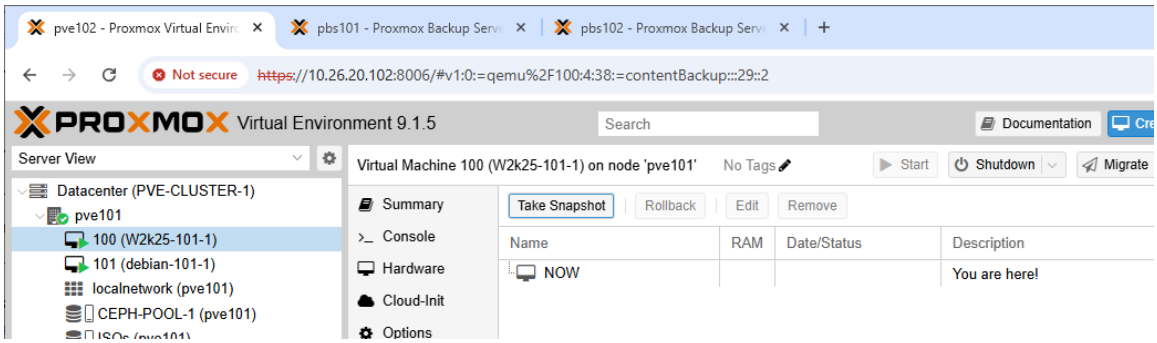
e.g.:


Confirm
Are you sure you want to remove entry 'After-Notepad_pp'
Yes
No

e.g.:


Confirm
Are you sure you want to remove entry 'Fresh-Install'
Yes
No

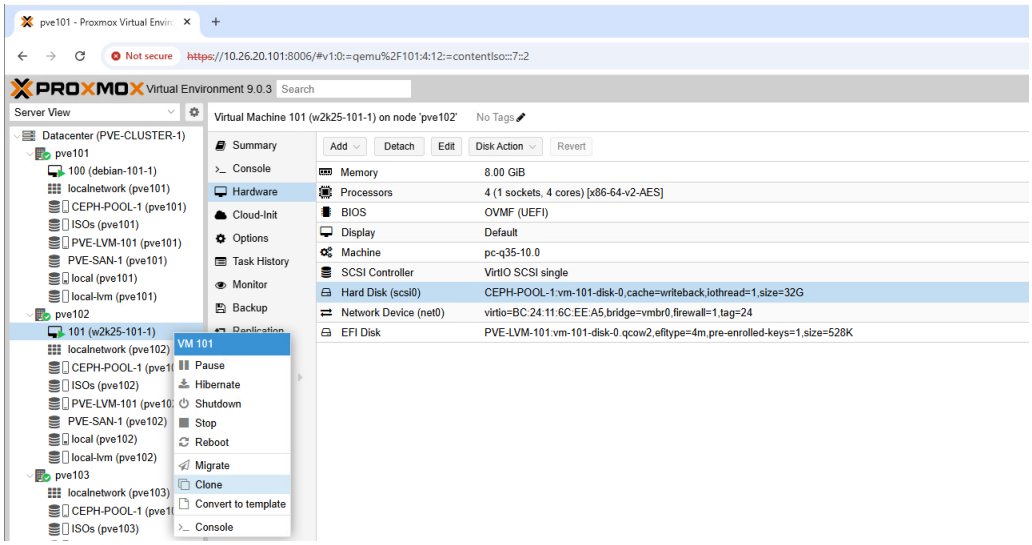
e.g.:


pve102 - Proxmox Virtual Envir
pbs101 - Proxmox Backup Serv
pbs102 - Proxmox Backup Serv
Not secure
https://10.26.20.102:8006/#v1:0:=qemu%2F100:4:38:=contentBackup:::29::2
PROXMOX Virtual Environment 9.1.5
Search
Documentation
Server View
Virtual Machine 100 (W2k25-101-1) on node 'pve101'
No Tags
Start
Shutdown
Migrate
Datacenter (PVE-CLUSTER-1)
pve101
100 (W2k25-101-1)
101 (debian-101-1)
localnetwork (pve101)
CEPH-POOL-1 (pve101)
Summary
Console
Hardware
Cloud-Init
Options
Take Snapshot
Rollback
Edit
Remove
Name
RAM
Date/Status
Description
NOW
You are here!

SBS LAB –(GUI) Cloning a VM

Step 3: Choose your w2k25-XYZ-1 VM > Right-click > Clone

e.g.:



Step 4: Target node: same as source

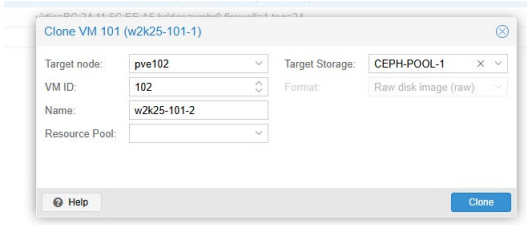
Step 5: VM ID: same as source

Step 6: Name: w2k25-XYZ-2

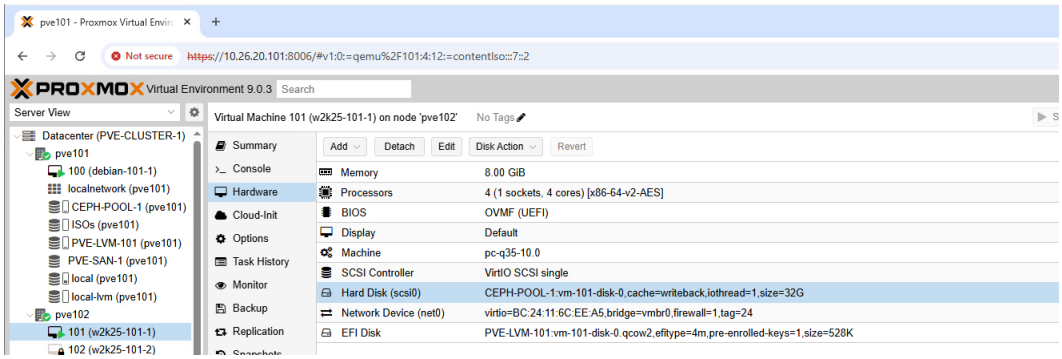
Step 7: Target Storage: same as source

Step 8: Clone

e.g.:



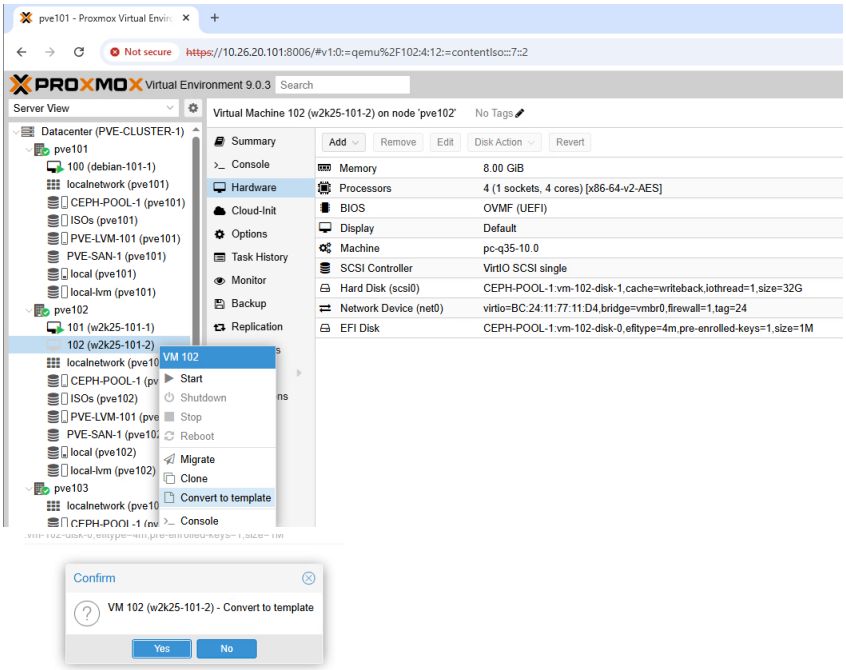
e.g.:



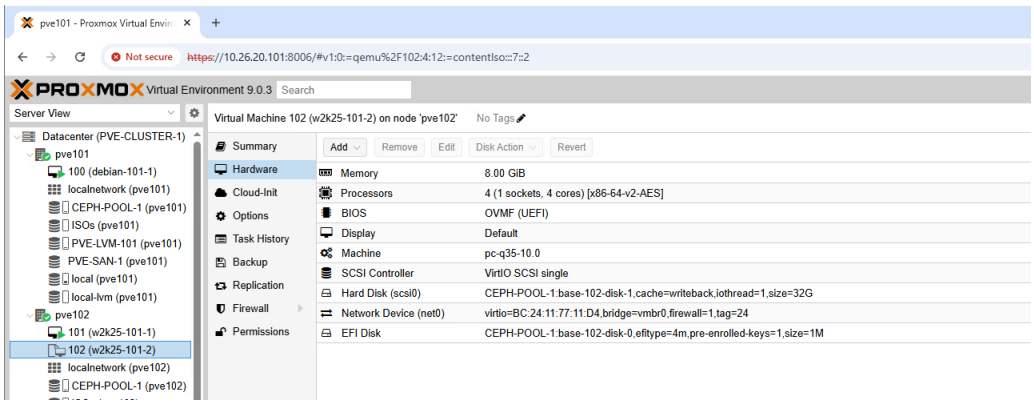
NOTE : Clone will be locked until complete

SBS LAB – (GUI) Creating a VM Template

Step 1: Right-click your w2k25-XYZ-2 VM > Convert to template



e.g.:

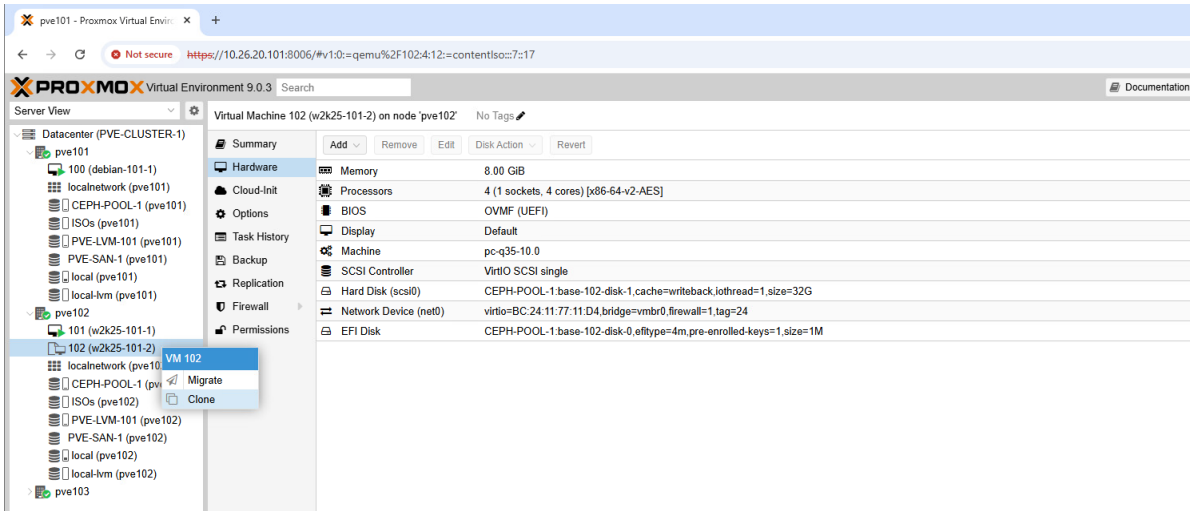


e.g.:

SBS LAB – (GUI) Deploy Linked Clone from Template

Step 1: Right-click your w2k25-XYZ-2 template > Clone

e.g.:



Step 2: Target node: same as source

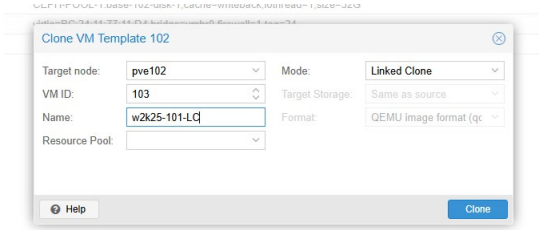
Step 3: VM ID: as automatically assigned

Step 4: Name: w2k25-XYZ-LC

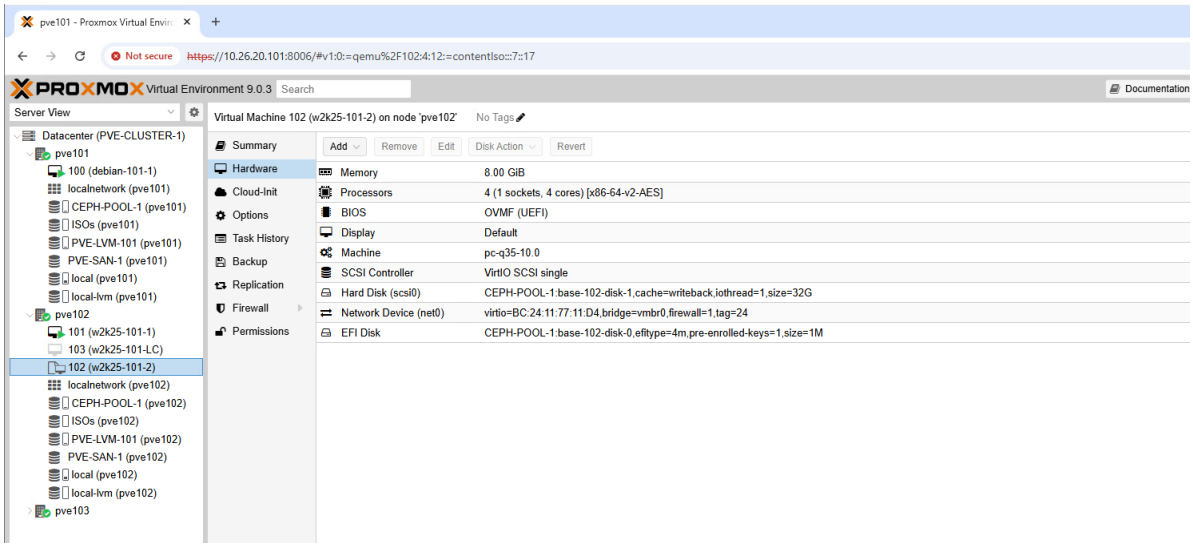
Step 5: Mode: Linked Clone

Step 6: Clone

e.g.:



e.g.:

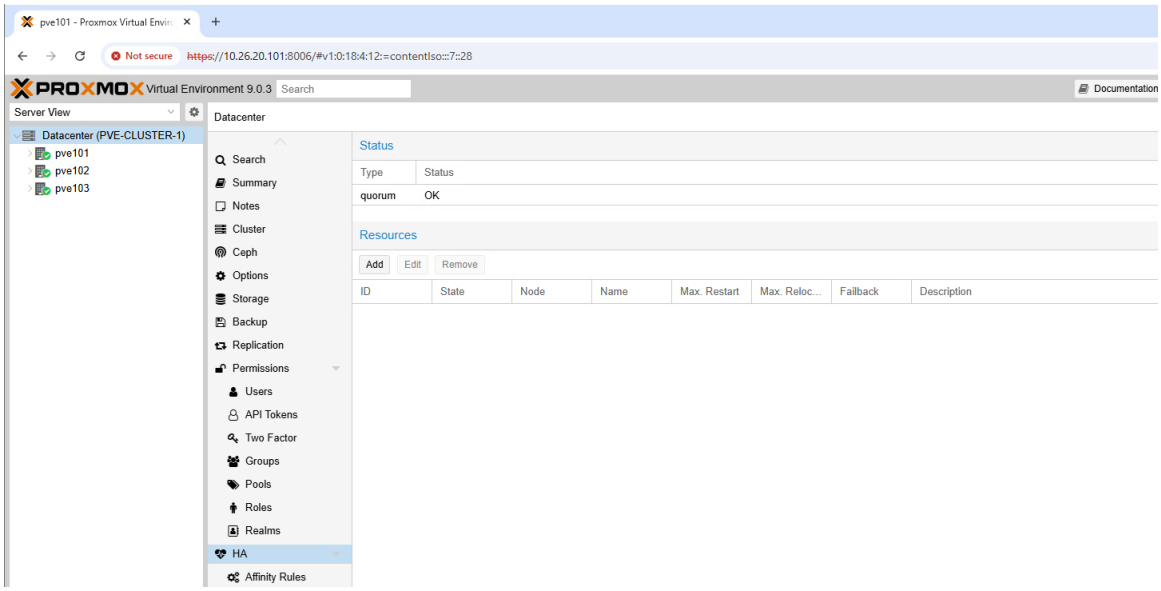


Resource Management and HA for PVE

SBS LAB –(GUI) HA for PVE

Step 1: Datacenter > HA > Resources > Add

e.g.:

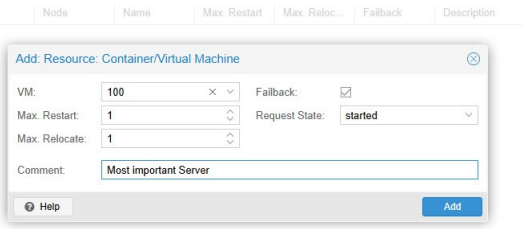


Step 2: Choose VM from dropdown

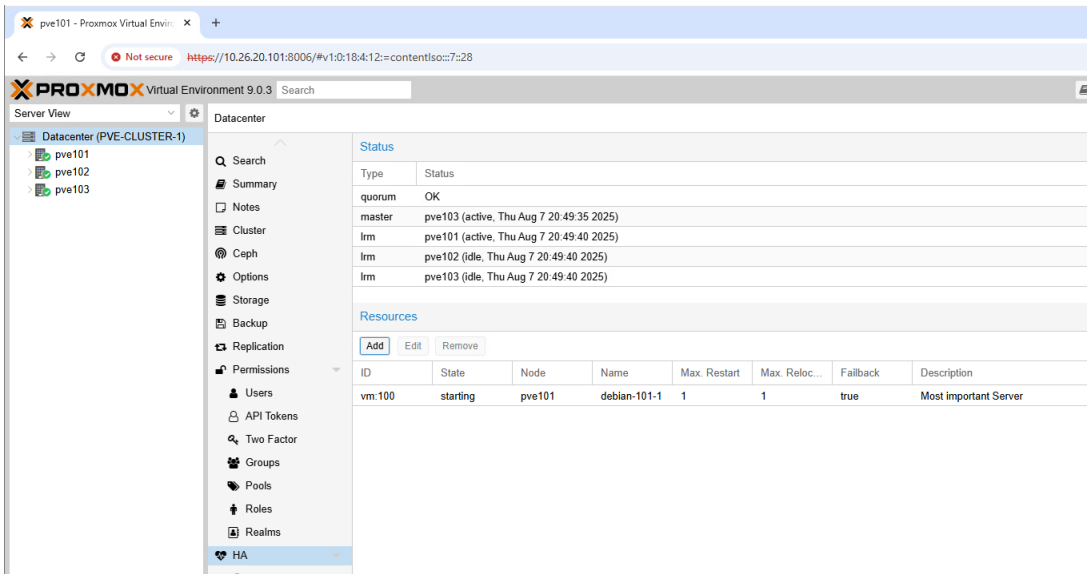
Step 3: Comment

Step 4: Add

e.g.:



Step 5: Datacenter > HA > Resources > Add

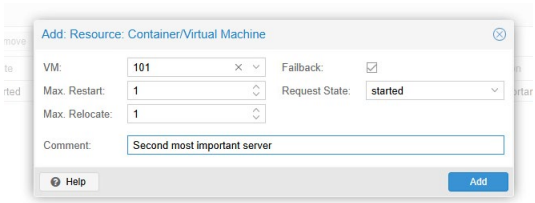


e.g.:

Step 6: Choose VM from dropdown

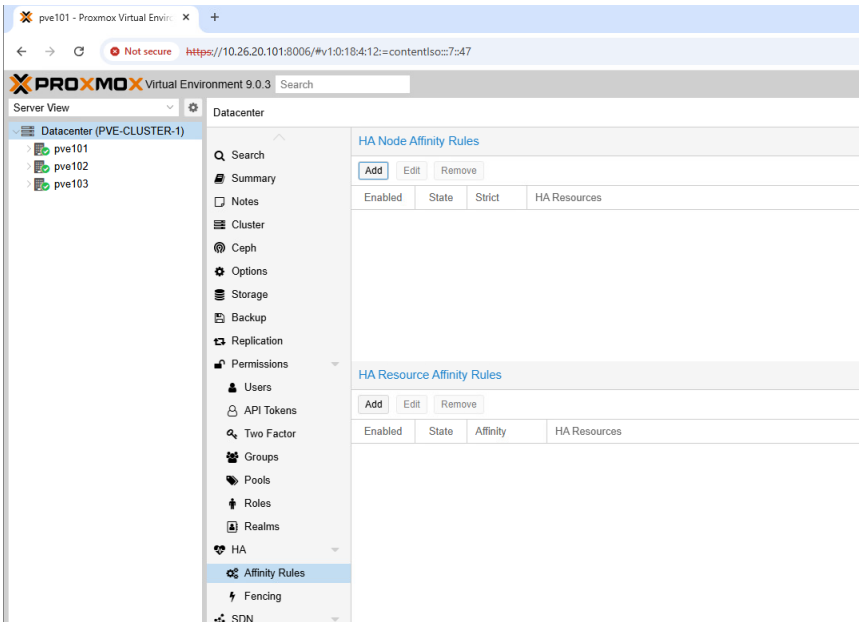
Step 7: Comment

Step 8: Add



e.g.:

Step 9: Datacenter > HA > Affinity Rules > HA Node Affinity Rules > Add



Step 10: Choose a VM (HA Resources)

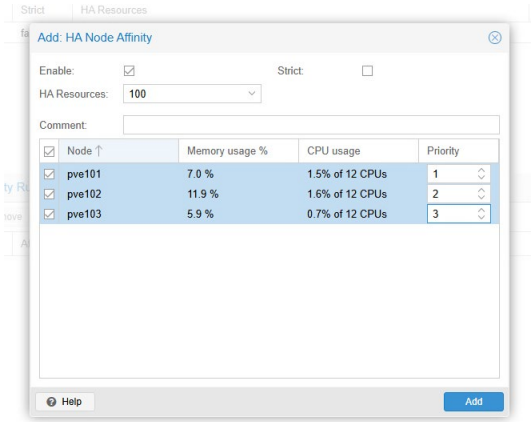
Step 11: Select all nodes where it can run

Step 12: Choose priority

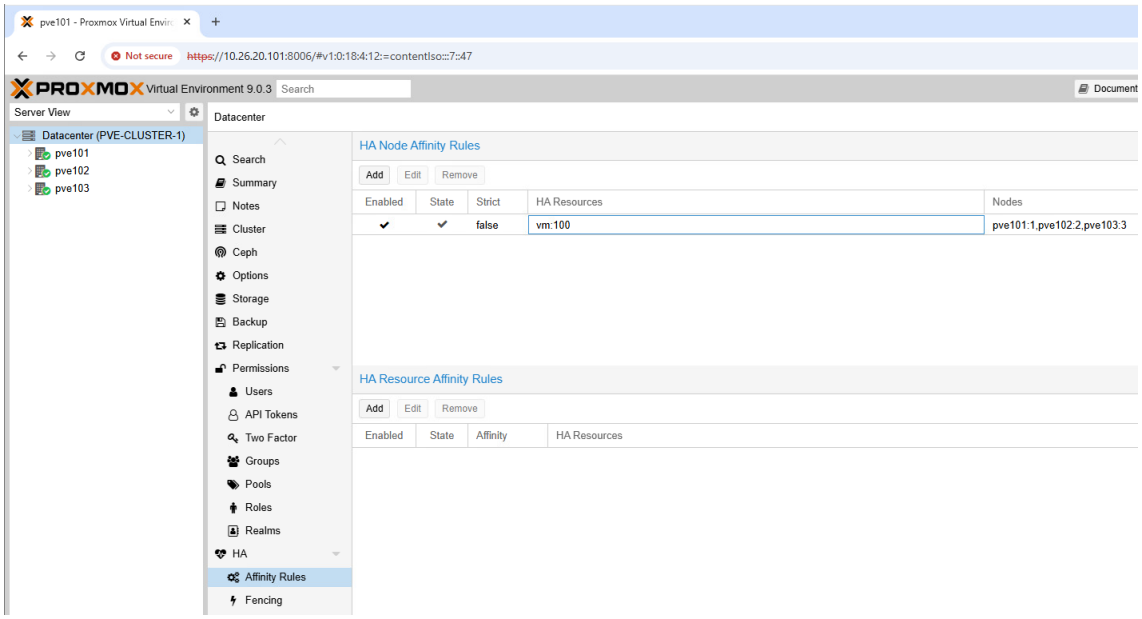
NOTE : If priority is same, there is no affinity

NOTE : Higher number = higher priority

Step 13: Add



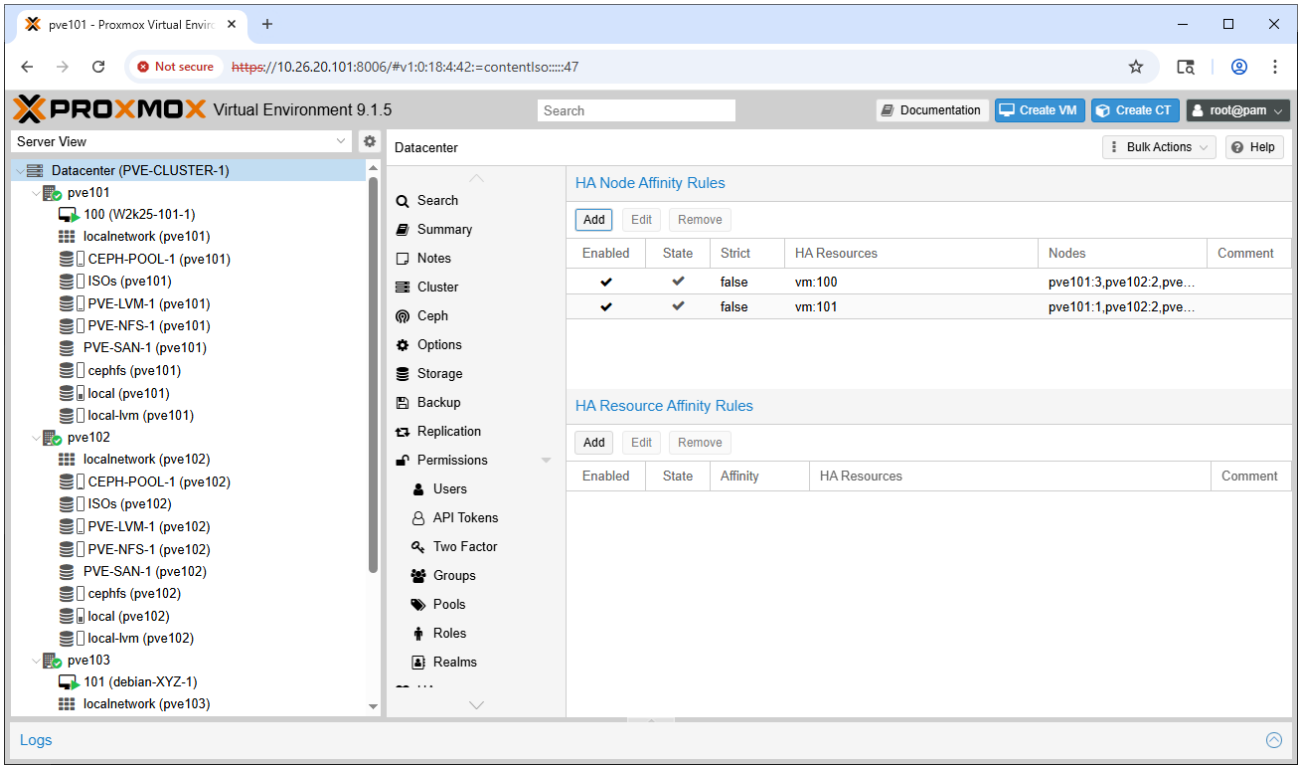
e.g.:



e.g.:

Step 14: Do this for all your VMs

e.g.:



SBS LAB – (GUI) Test HA Failover

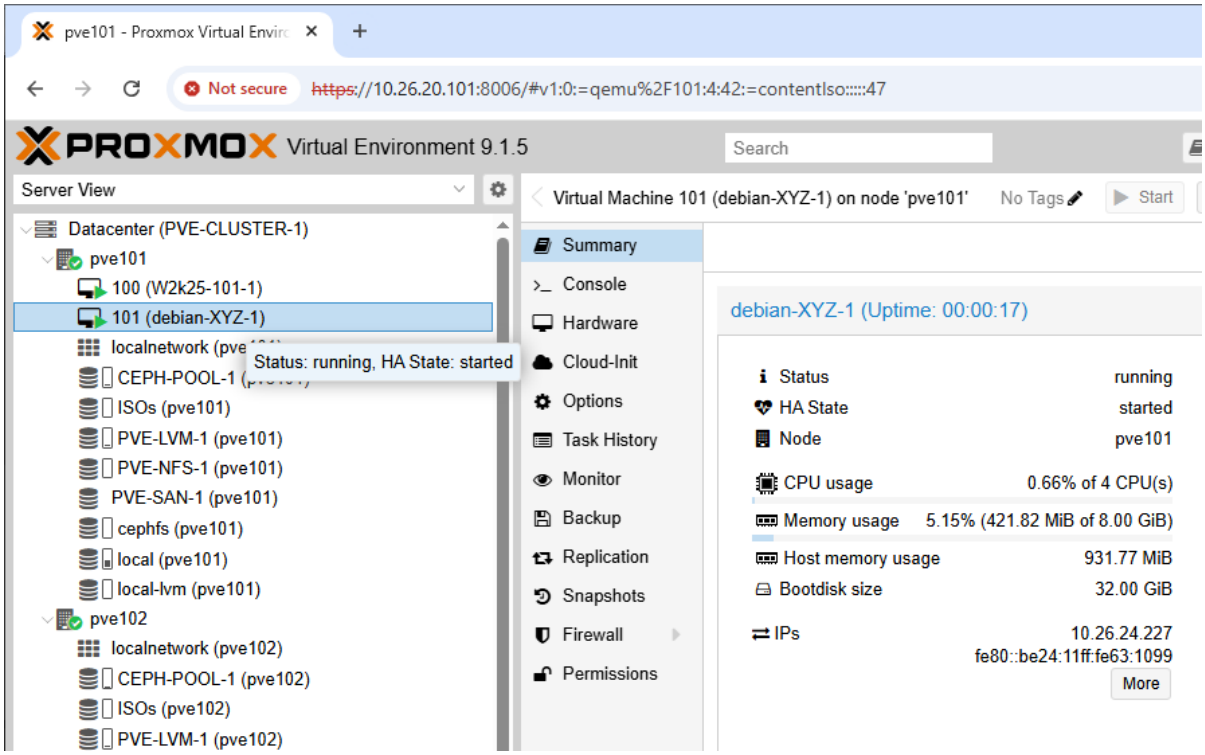
NOTE: If participating in a group environment, it is important that only one (possibly several in larger clusters) participant execute this lab. Instructor will designate participant.

Goals:

- Test HA failover
- Test integrity of Ceph cluster during/after loss of one node

Step 1: Determine the IP of one of your VMs

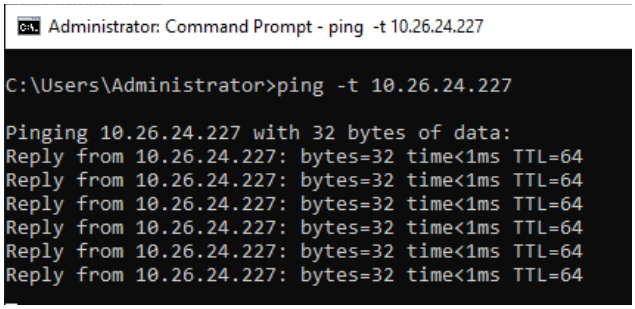
e.g.:



NOTE : If Windows and no ping received, disable Windows Firewall in VM

Step 2: Start a continuous ping from any other source on the network except HA VM

e.g.:

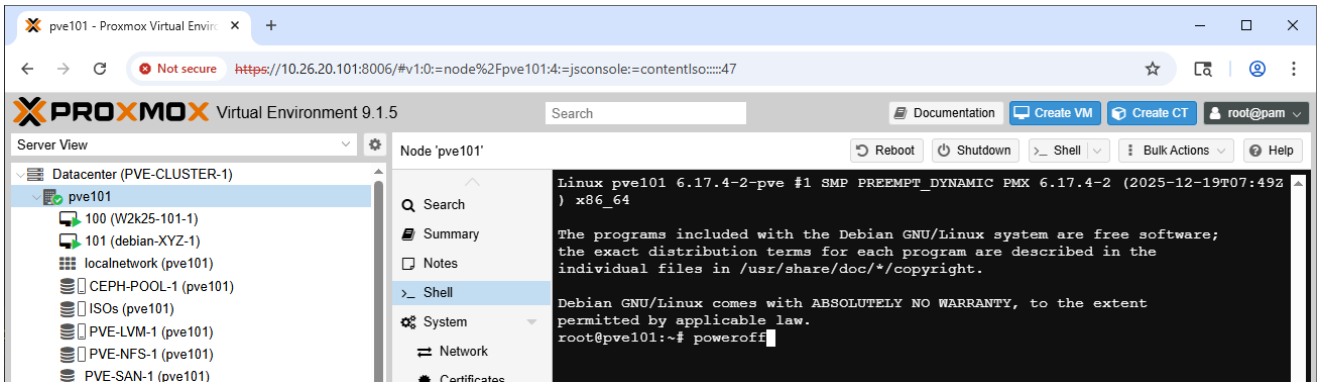


NOTE : We recommend starting the ping from your Windows Desktop

3. Power off PVE node

Step 1: pveXYZ > Shell > command: poweroff

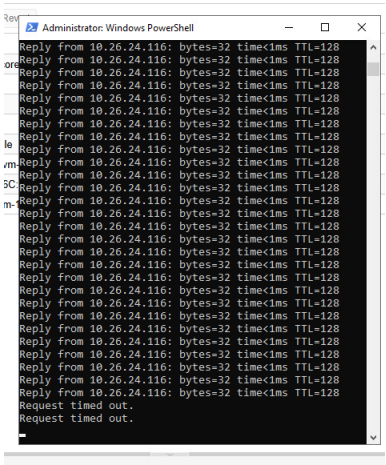
e.g.:



NOTE : If you are actually connected to the node that you shutdown, you will need to connect to a different node in the cluster to observe HA and re-connect to the console.

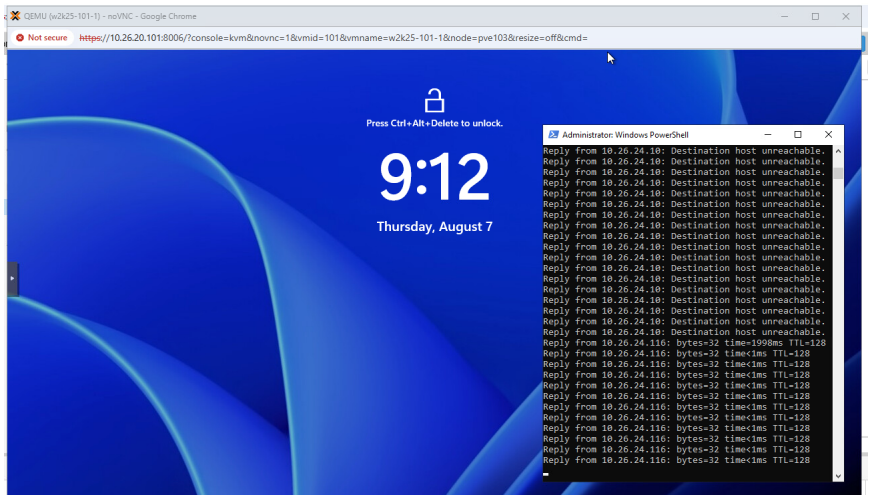
Step 2: Watch the ping

e.g.:



Step 3: Wait for HA to observe the console and ping to come back

e.g.:



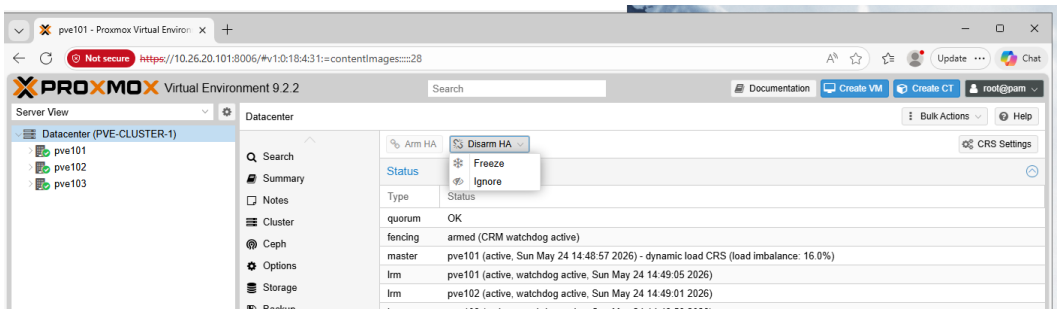
SBS LAB – (GUI) HA Arm/Disarm

When doing network maintenance or host updates, it might be possible to trigger HA inadvertently and cause the restart of critical workloads. For this reason, it is advisable to disarm HA prior to performing those tasks.

1. Disarm HA - Freeze

   Step 1: Datacenter > HA > Disarm

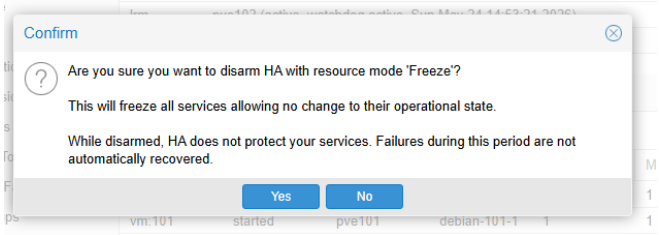
e.g.:



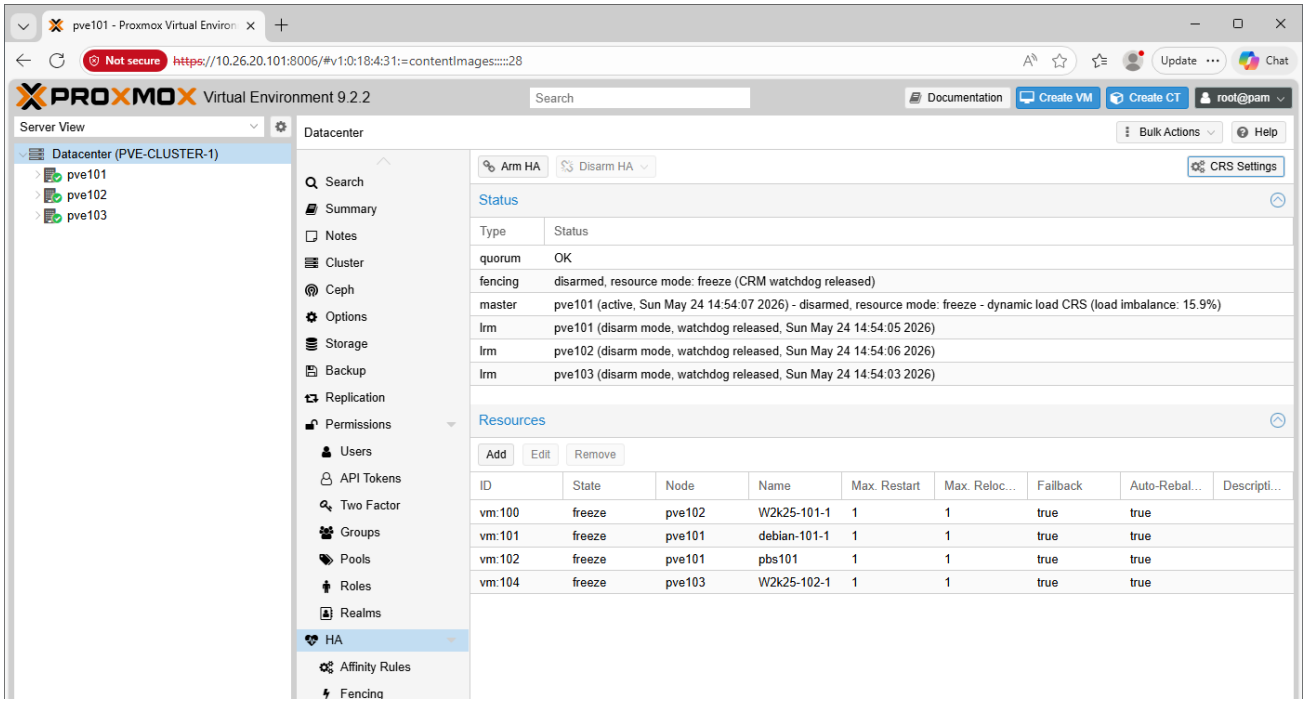
NOTE : Freeze stops HA monitoring without changing any resource constraints, allowing you to do maintenance safely

NOTE : Ignore mode allows manual resource re-allocation while enabled (such as migrating VMs) but will require manually restarting HA on resources to which it was applied previously

e.g.:



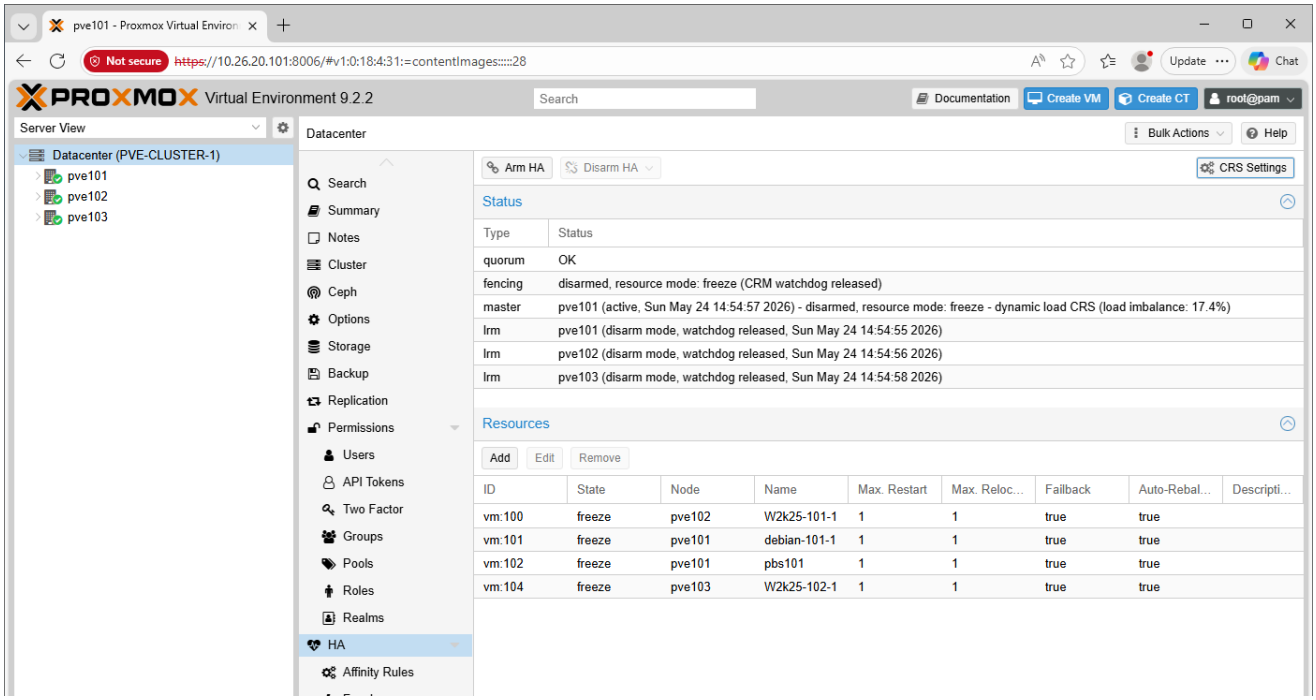
e.g.:



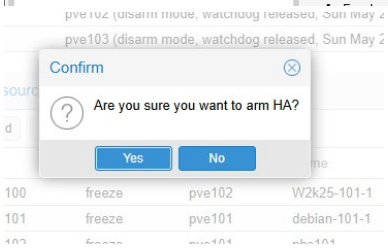
e.g.:

Step 2: When Maintenance is complete, Arm HA

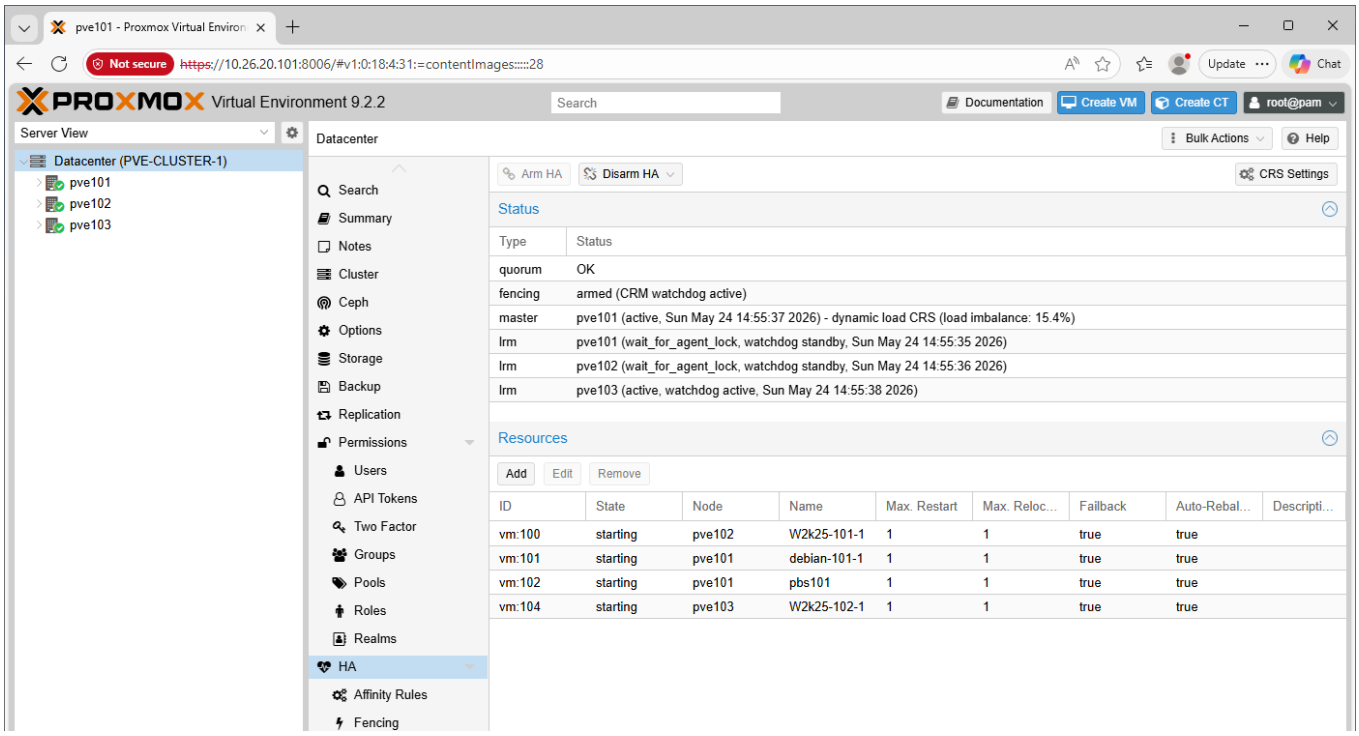
e.g.:



e.g.:



e.g.:

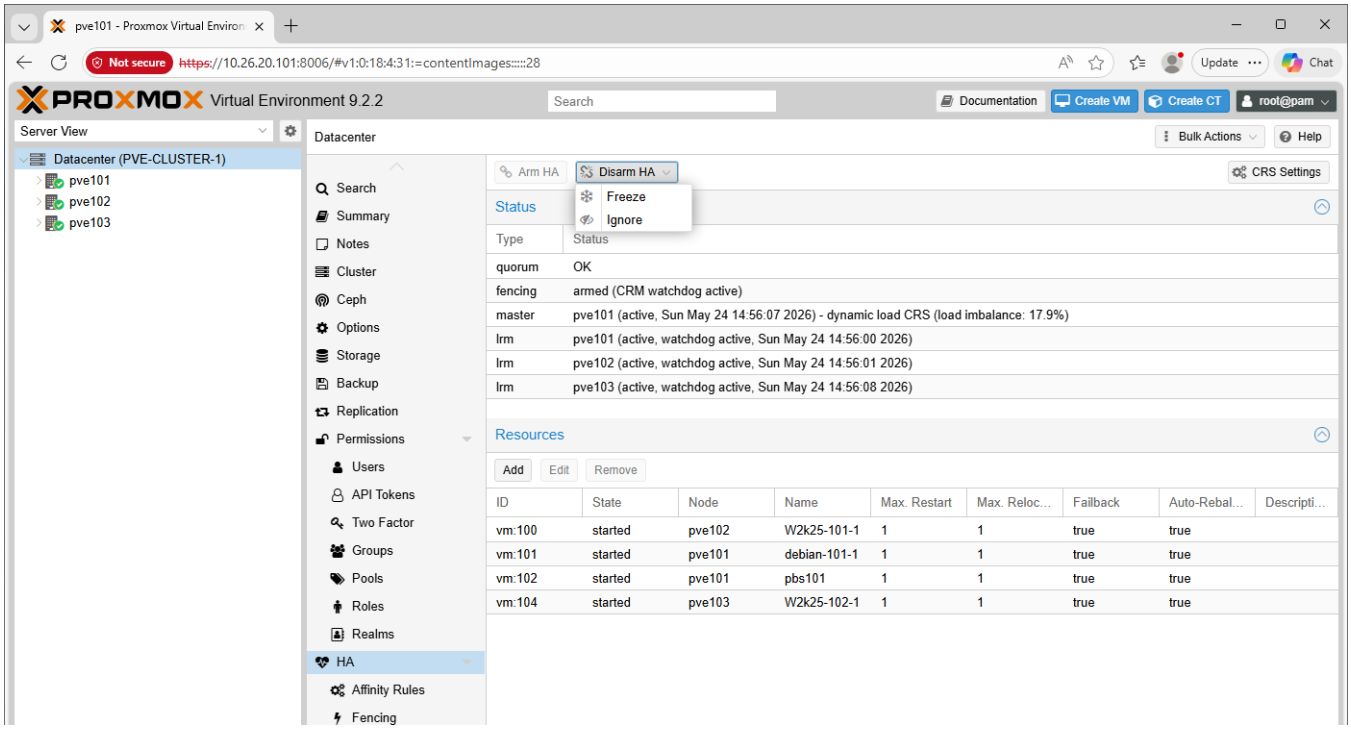


e.g.:

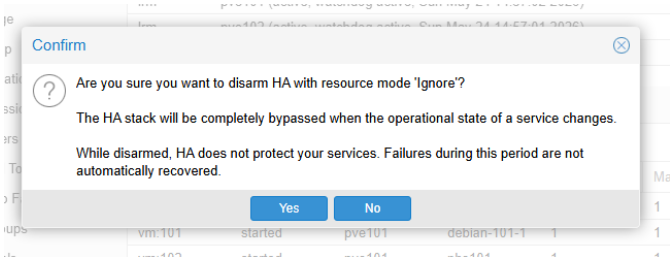
2. Disarm HA – Ignore

Step 1: Disarm HA – Ignore

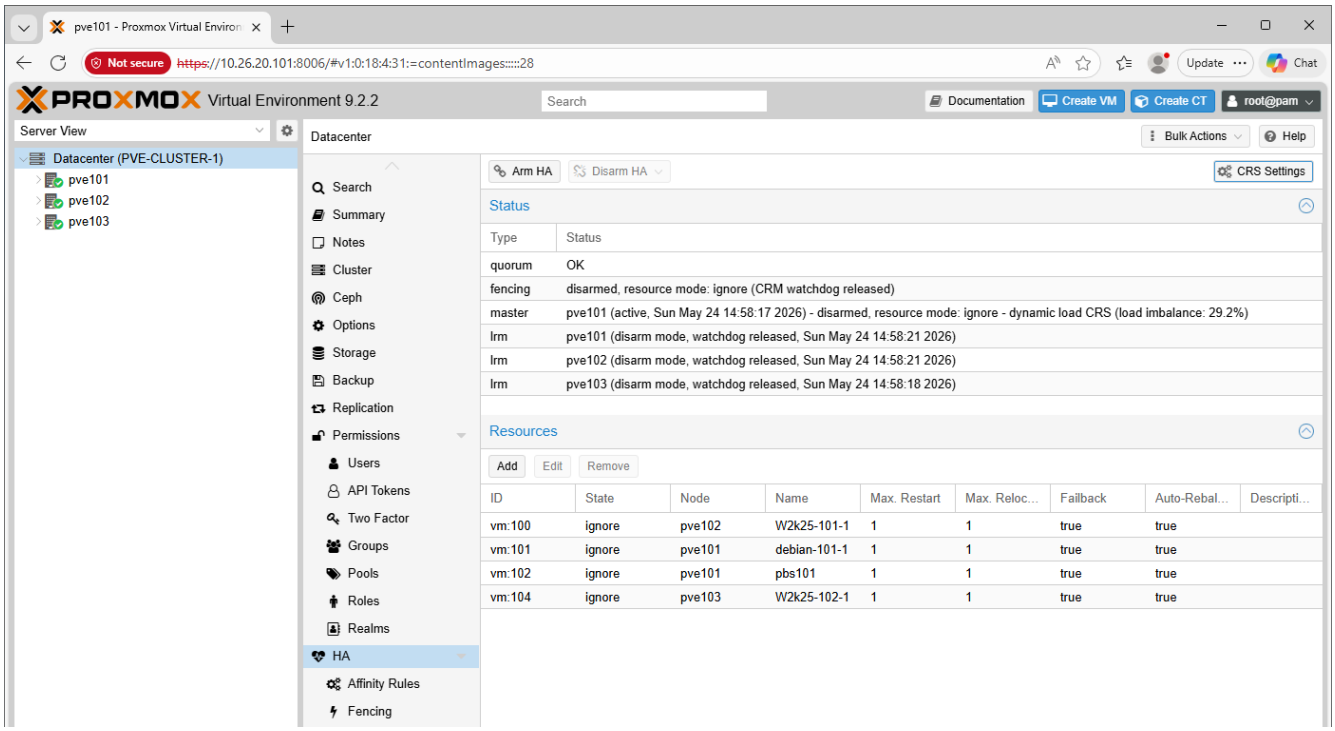
e.g.:



e.g.:

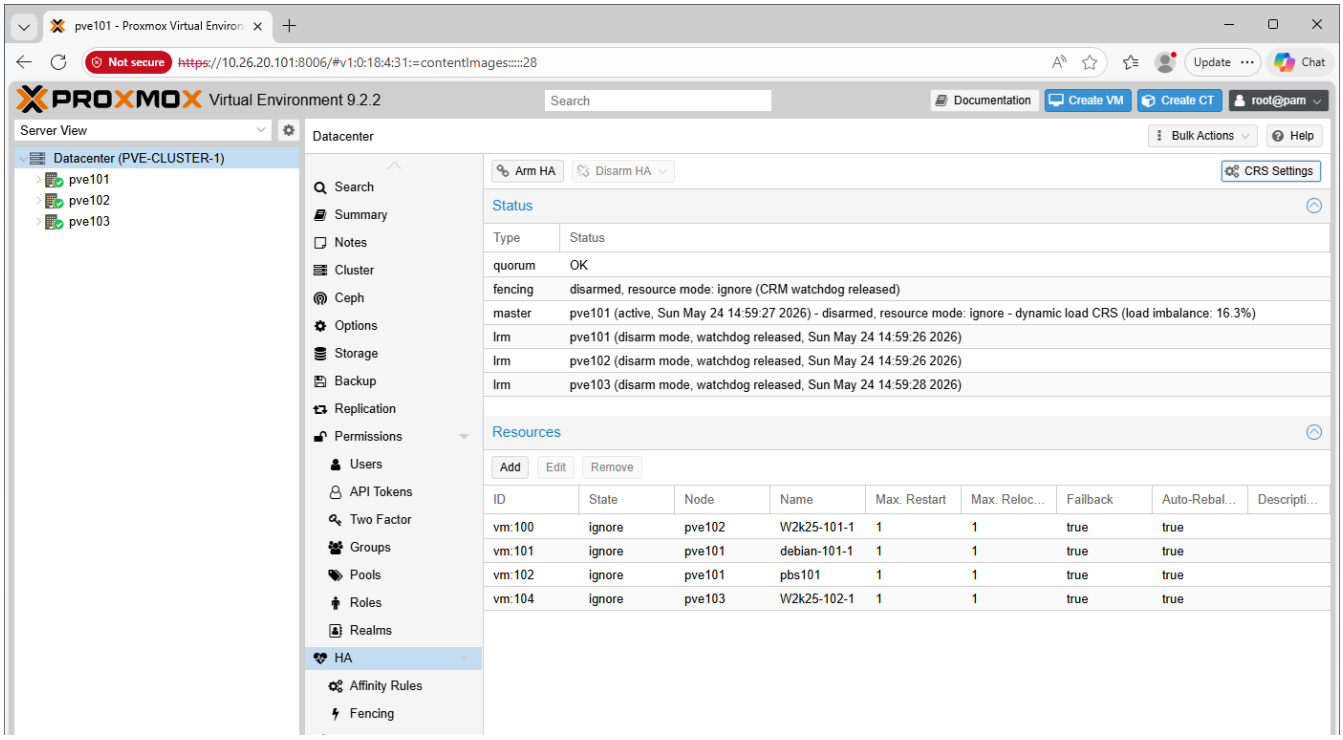


e.g.:

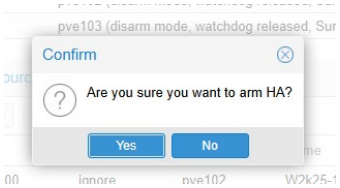


Step 2: When maintenance complete, Arm HA

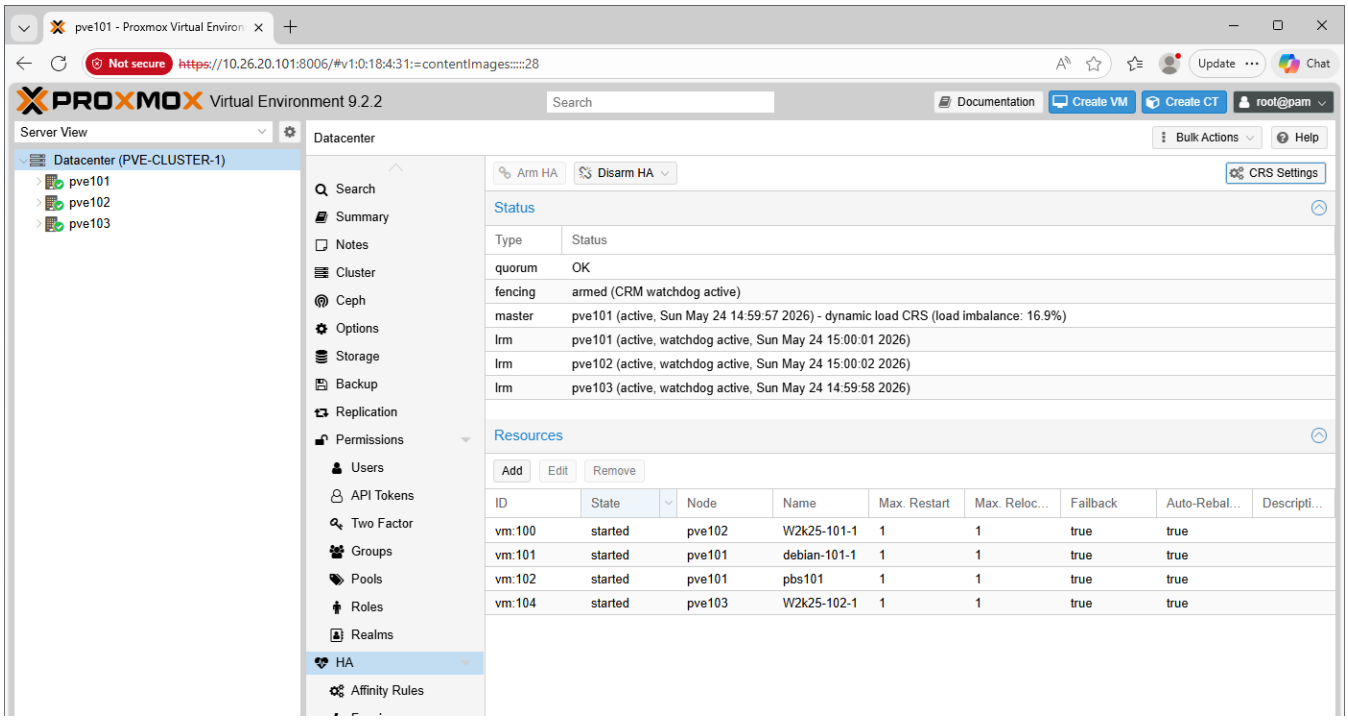
e.g.:



e.g.:

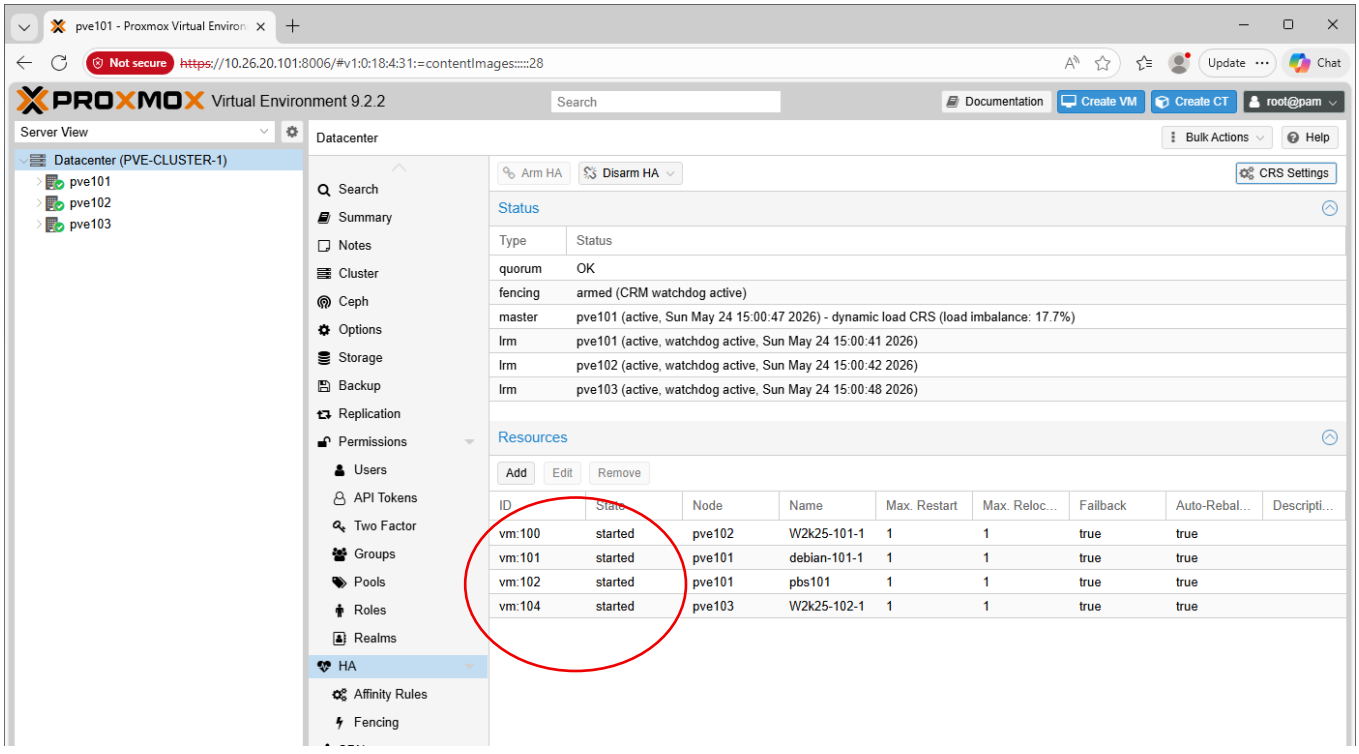


e.g.:



Step 3: Verify HA started for any modified resources

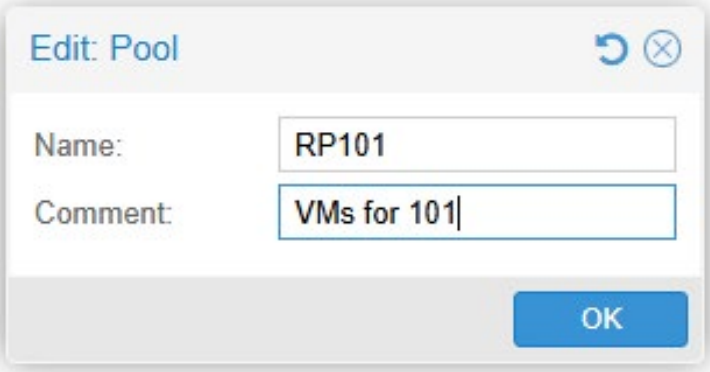
e.g.:



SBS LAB – (GUI) Resource Pools for PVE

In PVE a Resource Pool is primarily used to control access. It can contain VMs or Storage entities and then be assigned specific user/permissions

Step 4: s > Create
Step 5: Name: RPXYZ
Step 6: Comment: VMs for XYZ

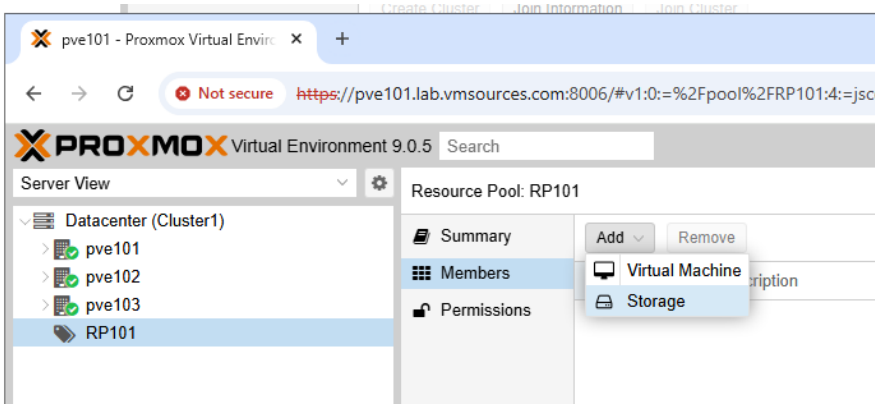


e.g.:

3. Move your VMs into RPXYZ

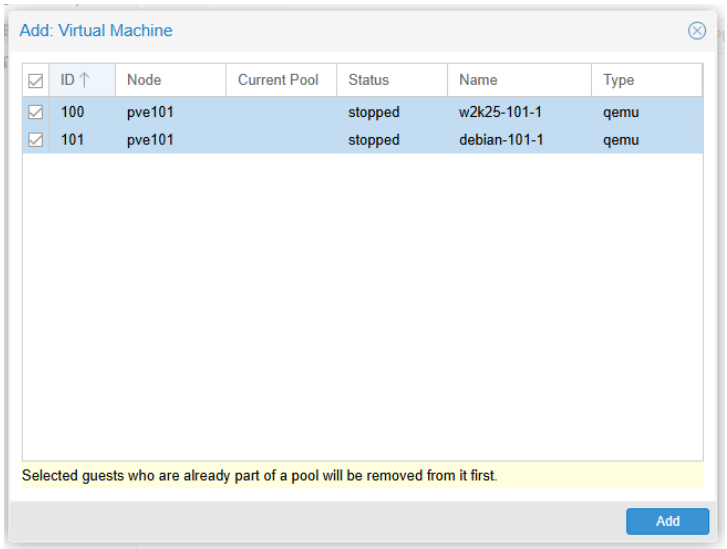
Step 1: RPXYZ > Add > Virtual Machine

e.g.:



Step 2: Select your VMs from the list > Add

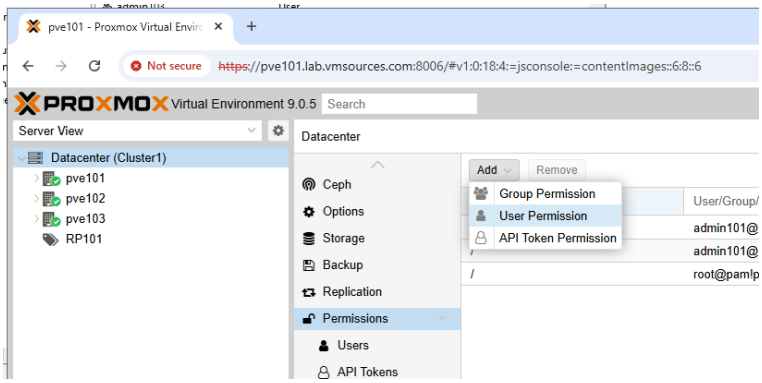
e.g.:



4. Now create a permission for a user to access the Resource Pool

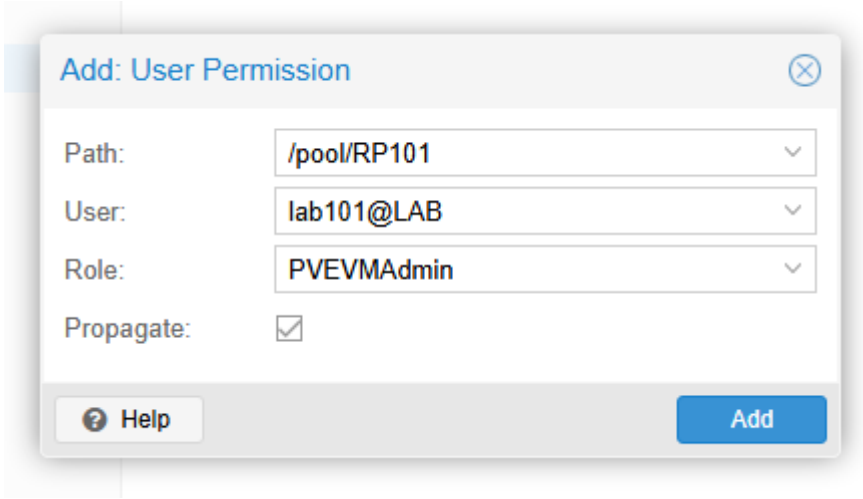
Step 1: Datacenter > Permissions > Add > User Permission

e.g.:



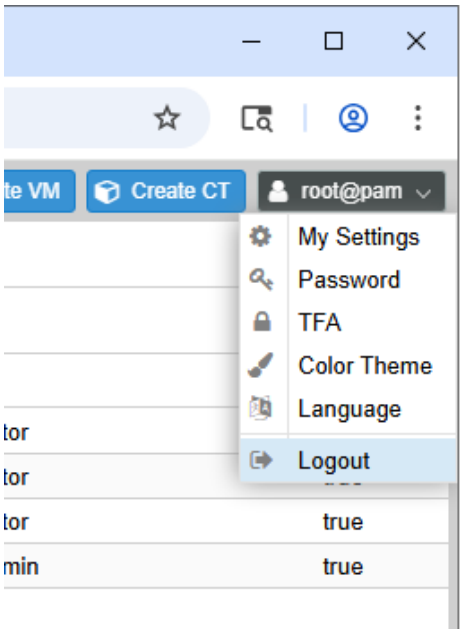
Step 2: Path: /pool/RPXYZ
Step 3: User: labXYZ@LAB
Step 4: Role: PVEVMAdmin
Step 5: Propagate: Checked

e.g.:

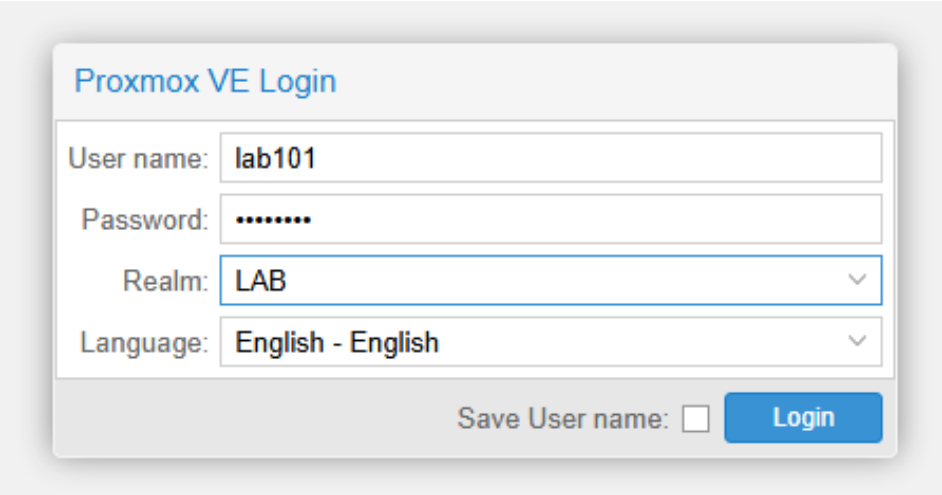


5. Log out and back in as labXYZ@LAB

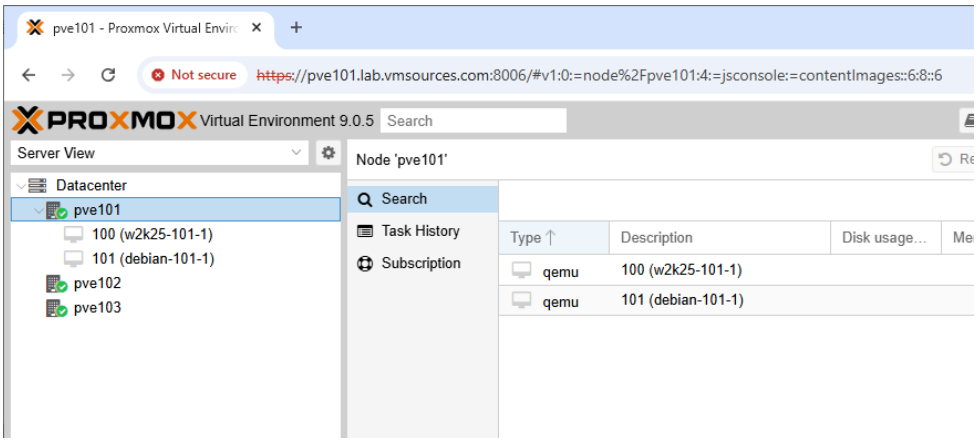
Step 1:



Step 2:


Proxmox VE Login
User name:
lab101
Password:
Realm:
LAB
Language:
English - English
Save User name:
Login

Step 3:


pve101 - Proxmox Virtual Envir
Not secure
https://pve101.lab.vmsources.com:8006/#v1:0:=node%2Fpve101:4:=jsconsole:=contentImages::6:8::6
PROXMOX Virtual Environment 9.0.5
Search
Server View
Datacenter
pve101
100 (w2k25-101-1)
101 (debian-101-1)
pve102
pve103
Node 'pve101'
Search
Task History
Subscription
Type
Description
Disk usage...
qemu
100 (w2k25-101-1)
qemu
101 (debian-101-1)

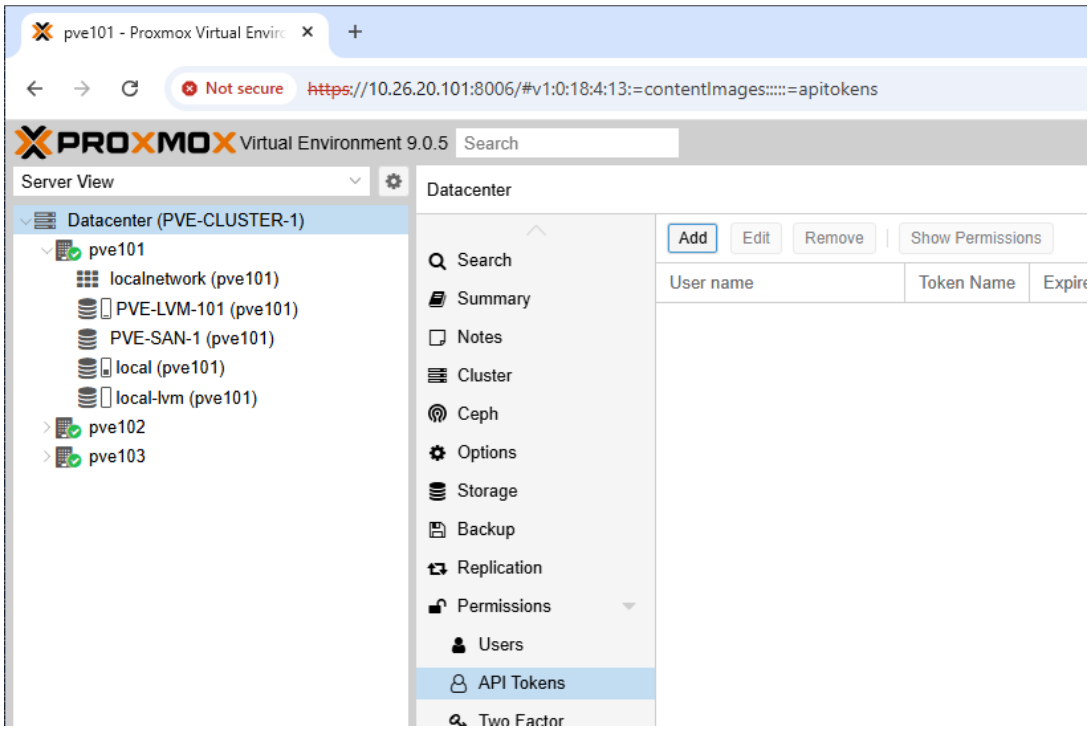
SBS LAB – (GUI) API Tokens for PVE

API Tokens provide a more secure form of authentication for applications and services which need to access PVE.

1. Generating an API Token for ProxLB to use for authentication

   Step 1: Go to: Datacenter > Permissions > API Tokens > Add

   e.g.:



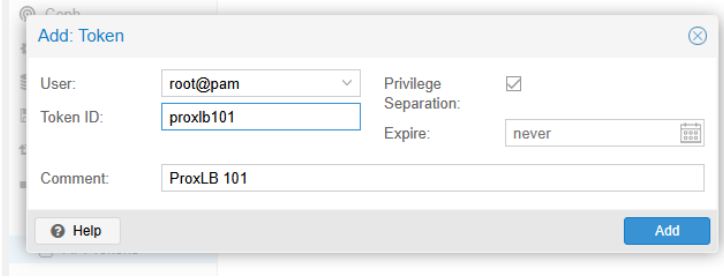
   Step 2: User: root@pam

   Step 3: Token ID: proxlbXYZ

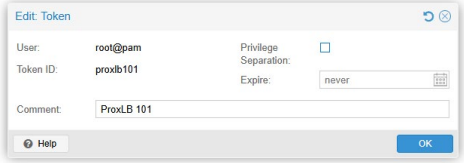
   Step 4: Comment: ProxLB XYZ

   Step 5: Add

   e.g.:

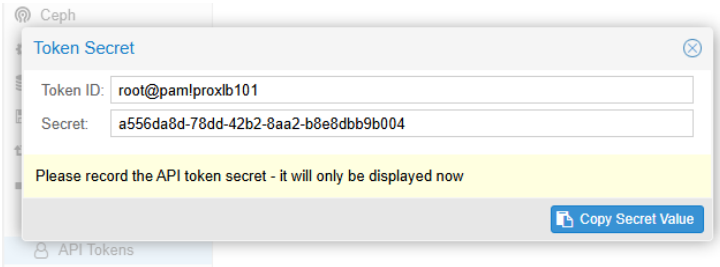


   e.g.:



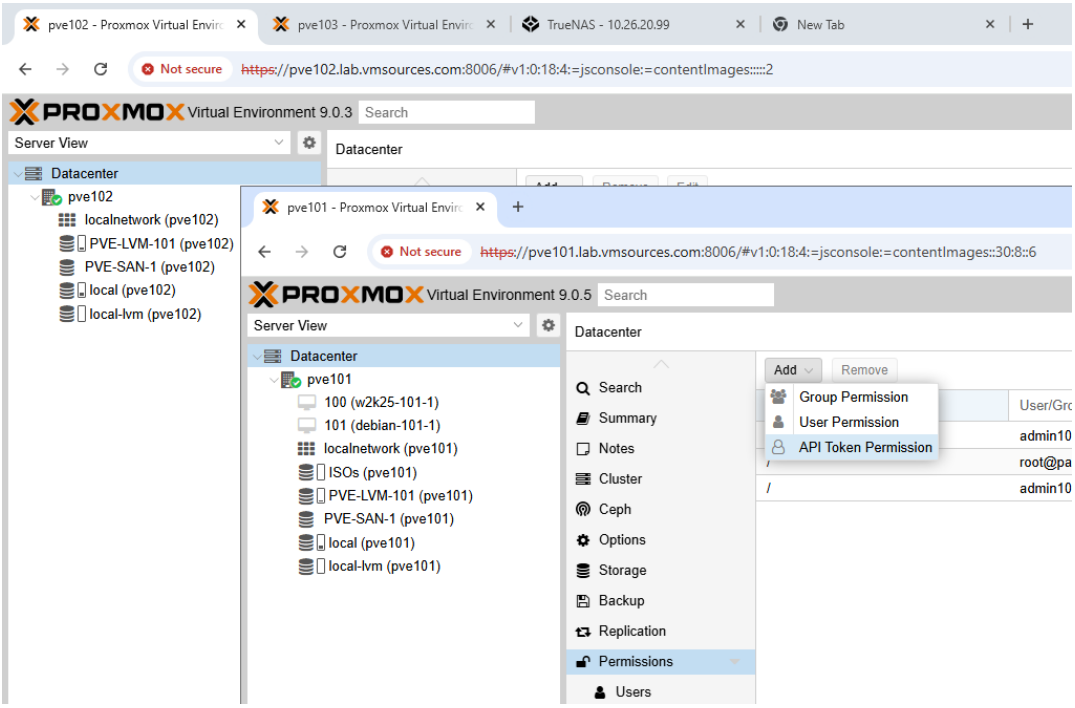
Step 6: Copy and record the secret value, it can not be retrieved later

e.g.:



Step 7: Datacenter > Permissions > Add > API Token Permission

e.g.:



Step 8: Path: /

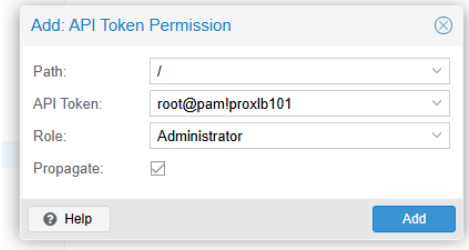
Step 9: API Token: root@pam!proxlb

Step 10: Rols: Administrator

Step 11: Propagate: Checked

Step 12: Add

e.g.:



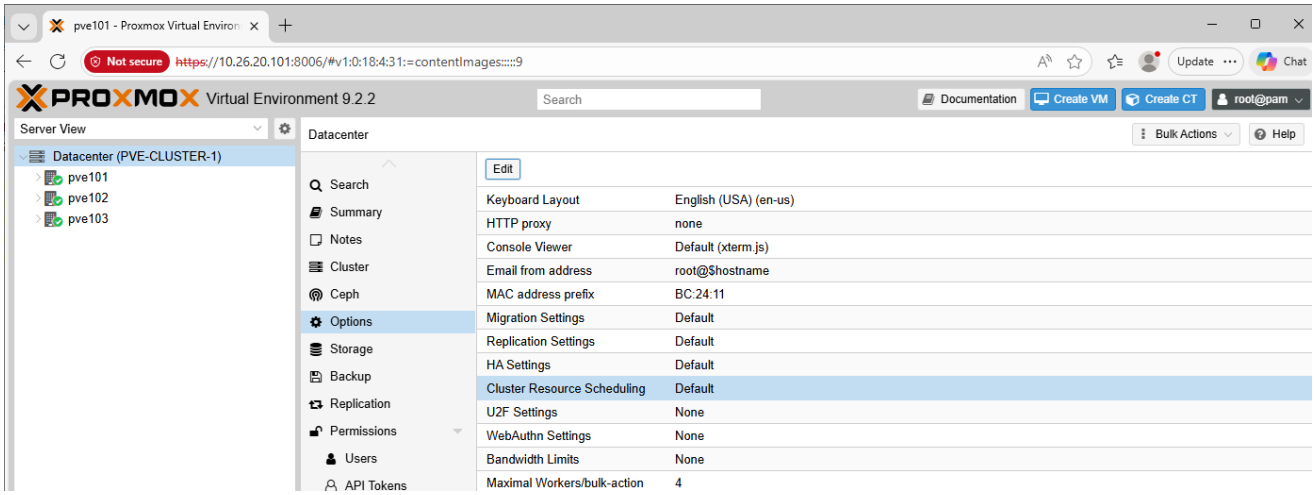
SBS LAB – (GUI) Dynamic Load Balancer for Proxmox

Native Dynamic Load Balancing (DLB) for PVE was introduced with version 9.2

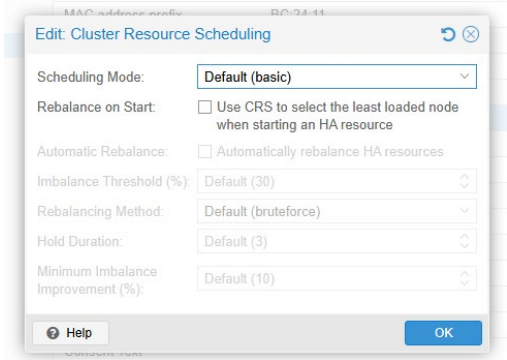
1. Configure DLB

Step 1: Datacenter > Options > Cluster Resource Scheduling > Edit

e.g.:

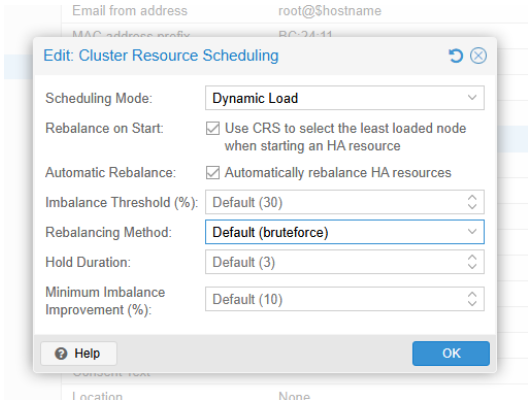


e.g.:



Step 2: Select Dynamic Load and choose your options

e.g.:

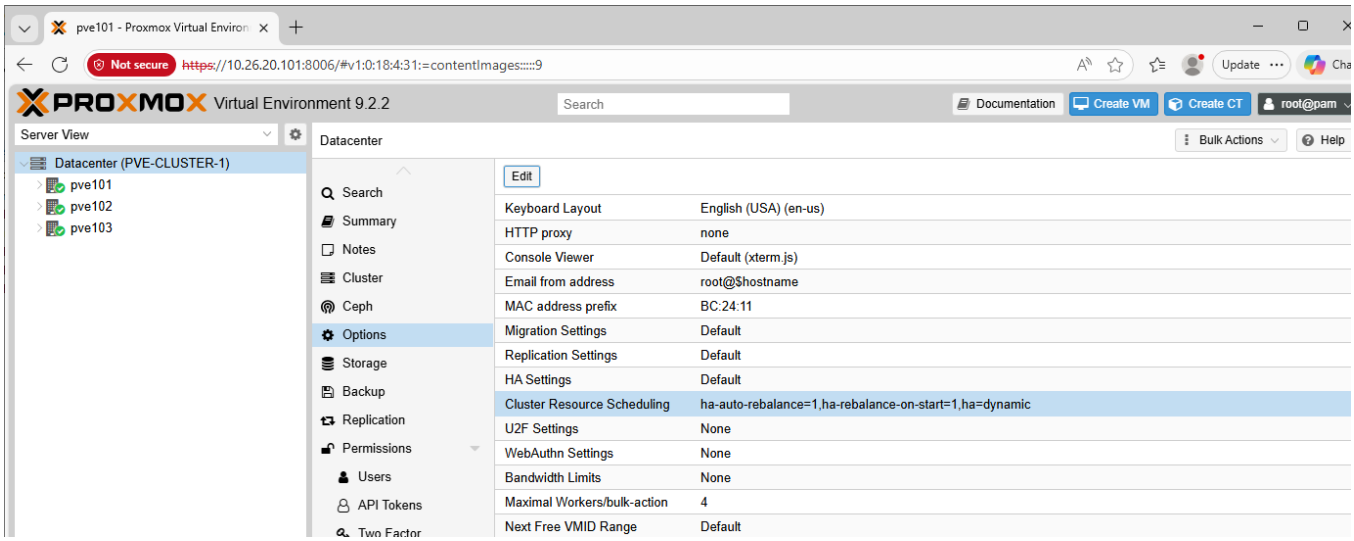


NOTE : Rebalancing Method Default (bruteforce) weighs RAM and CPU utilization equally

NOTE : Rebalancing Method TOPSYS weighs RAM as higher priority with a 5:1 priority for average utilization and a 10:5 ratio for peak usage

(i) https://forum.proxmox.com/threads/questions-about-the-dynamic-crs.183171/

e.g.:



SBS LAB – (CLI) ProxLB Active Load Balancing for PVE (deprecated)

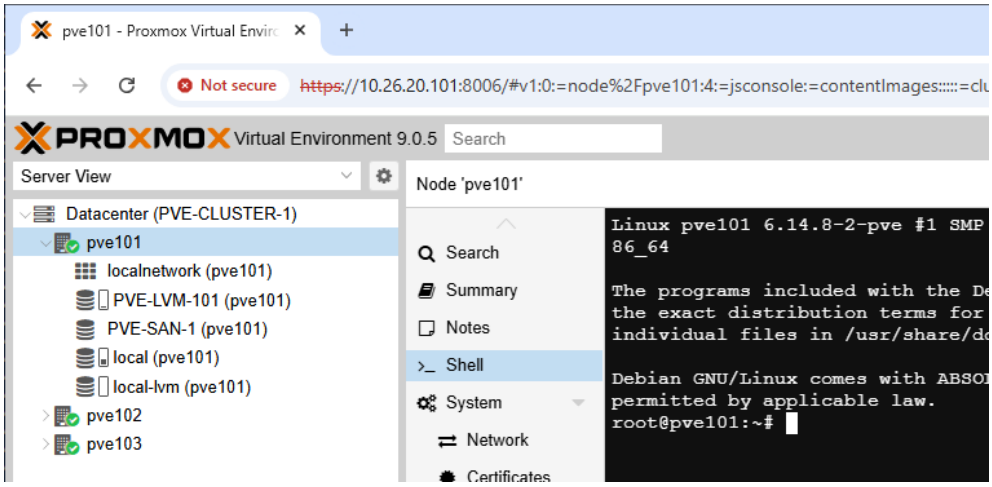
NOTE: This lab is deprecated for users of Proxmox 9.2 as Dynamic Load Balancer is included natively

ProxLB is a tool created by gyptazy and maintained on GitHub. It is fully open-source (GPLv3) and widely used and respected by the PVE community. Since it is open-source, community review is likely to have discovered and reported any known security flaws in the ProxLB architecture, so it is also regarded to be a safe addition to PVE.

2. Now we'll install ProxLB

   Step 1: Open a shell to your pveXYZ node

   e.g.:



   Step 2: Add the repos to the node

   Command #1 run: echo "deb https://repo.gyptazy.com/stable /" > /etc/apt/sources.list.d/proxlb.list

   e.g.:

```
Linux pve101 6.14.8-2-pve #1 SMP PREEMPT_DYNAMIC PMX 6.14.8-2 (2025-07-22T10:04Z) x
86_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.
root@pve101:~# echo "deb https://repo.gyptazy.com/stable /" > /etc/apt/sources.list
.d/proxlb.list
root@pve101:~#
```

Step 3: Get the encryption keys for the new repo

Command #1 run: wget -O /etc/apt/trusted.gpg.d/proxlb.asc https://repo.gyptazy.com/repository.gpg

e.g.:

```
.d/proxlb.list
root@pve101:~# wget -O /etc/apt/trusted.gpg.d/proxlb.asc https://repo.gyptazy.com/r
epository.gpg
--2025-08-18 16:15:56--  https://repo.gyptazy.com/repository.gpg
Resolving repo.gyptazy.com (repo.gyptazy.com)... 77.90.10.90, 2a13:e3c1:1337::c0de
Connecting to repo.gyptazy.com (repo.gyptazy.com)|77.90.10.90|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3923 (3.8K) [application/octet-stream]
Saving to: '/etc/apt/trusted.gpg.d/proxlb.asc'

/etc/apt/trusted.gpg 100%[===================>]   3.83K  --.-KB/s    in 0s

2025-08-18 16:16:01 (79.1 MB/s) - '/etc/apt/trusted.gpg.d/proxlb.asc' saved [3923/3
923]

root@pve101:~#
```

Step 4: Update repo metadata and install ProxLB

Command #1 run: apt-get update && apt-get -y install proxlb

e.g.:

```
923]

root@pve101:~# apt-get update && apt-get -y install proxlb
```

e.g.:

```
Selecting previously unselected package libsodium23:amd64.
(Reading database ... 53721 files and directories currently installed.)
Preparing to unpack .../0-libsodium23_1.0.18-1+b2_amd64.deb ...
Unpacking libsodium23:amd64 (1.0.18-1+b2) ...
Selecting previously unselected package python3-proxmoxer.
Preparing to unpack .../1-python3-proxmoxer_2.2.0-1_all.deb ...
Unpacking python3-proxmoxer (2.2.0-1) ...
Selecting previously unselected package proxlb.
Preparing to unpack .../2-proxlb_1.1.5_all.deb ...
Unpacking proxlb (1.1.5) ...
Selecting previously unselected package python3-invoke.
Preparing to unpack .../3-python3-invoke_2.2.0-2_all.deb ...
Unpacking python3-invoke (2.2.0-2) ...
Selecting previously unselected package python3-nacl.
Preparing to unpack .../4-python3-nacl_1.5.0-7_amd64.deb ...
Unpacking python3-nacl (1.5.0-7) ...
Selecting previously unselected package python3-paramiko.
Preparing to unpack .../5-python3-paramiko_3.5.1-3_all.deb ...
Unpacking python3-paramiko (3.5.1-3) ...
Setting up libsodium23:amd64 (1.0.18-1+b2) ...
Setting up python3-proxmoxer (2.2.0-1) ...
Setting up python3-invoke (2.2.0-2) ...
Setting up python3-nacl (1.5.0-7) ...
Setting up proxlb (1.1.5) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/proxlb.service' → '/et
c/systemd/system/proxlb.service'.
User 'plb' created.
Setting up python3-paramiko (3.5.1-3) ...
Processing triggers for libc-bin (2.41-12) ...
root@pve101:~#
```

Step 5: Copy the example config to a working config

Command #1 run: cp /etc/proxlb/proxlb_example.yaml /etc/proxlb/proxlb.yaml

e.g.:

```
Processing triggers for libc-bin (2.41-12) ...
root@pve101:~# cp /etc/proxlb/proxlb_example.yaml /etc/proxlb/proxlb.yaml
root@pve101:~#
```

3. API Method

Step 1: Edit the config file

Command #1 run: vi /etc/proxlb/proxlb.yaml

e.g.:

```
proxmox_api:
  hosts: ['virt01.example.com', '10.10.10.10', 'fe01::bad:code::cafe']
  user: root@pam
  pass: crazyPassw0rd!
  # API Token method
  # token_id: proxlb
  # token_secret: 430e308f-1337-1337-beef-1337beefcafe
  ssl_verification: True
  timeout: 10
  # API Connection retries
  # retries: 1
  # wait_time: 1

proxmox_cluster:
  maintenance_nodes: ['virt66.example.com']
  ignore_nodes: []
  overprovisioning: True

balancing:
  enable: True
  enforce_affinity: False
  parallel: False
  # If running parallel job, you can define
  # the amount of prallel jobs (default: 5)
  parallel_jobs: 1
  live: True
  with_local_disks: True
  balance_types: ['vm', 'ct']
  max_job_validation: 1800
"/etc/proxlb/proxlb.yaml" 43 lines, 892 bytes
```

Step 2: Add ALL the PVE nodes in the cluster to the line ‘hosts’
Step 3: Comment out the pass line
Step 4: Uncomment sections relevant to the API
Step 5: Replace the Token ID with: proxlbXYZ
Step 6: Paste the Token you recorded in step 1
Step 7: Uncomment: retries
Step 8: Uncomment: wait_time

e.g.:

```
proxmox_api:
  hosts: ['pve101.lab.vmsources.com']
  user: root@pam
  #pass: P@ssw0rd
  # API Token method
  token_id: proxlb101
  token_secret: a06bfb14-ddbb-42b1-bb34-0801565d5dd6
  ssl_verification: False
  timeout: 10
  # API Connection retries
  retries: 1
  wait_time: 1
```

4. PAM Authentication (without API – do this only if you choose not to use API token)

e.g.:

```
proxmox_api:
  hosts: ['pve101.lab.vmsources.com']
  user: root@pam
  pass: P@ssw0rd
  # API Token method
  # token_id: proxlb
  # token_secret: 430e308f-1337-1337-beef-1337beefcafe
  ssl_verification: False
  timeout: 10
  # API Connection retries
  # retries: 1
  # wait_time: 1
```

5. Now, modify proxmox_cluster section
   Step 1: Delete any values from maintenance_nodes
   Step 2: add ‘master_only: 1’
   e.g.:

```
proxmox_cluster:
  maintenance_nodes: ['']
  ignore_nodes: []
  overprovisioning: True
  master_only: 1
```

   NOTE : The value ‘master_only: 1’ causes the Master HA node to be the active manager while ProxLB is still installed and available on other nodes.

6. Now modify the service section
   Step 1: interval: 5
   Step 2: format: minutes
   e.g.:

```
service:
  daemon: True
  schedule:
    interval: 5
    format: minutes
  delay:
    enable: False
    time: 1
    format: hours
  log_level: INFO
```

   NOTE : We do not suggest that these are appropriate values for your environment, however in a test lab, we do not want to wait forever to see actions take place.

7. Enable and start services
   Step 1: Enable and start the ProxLB service
   Command #1 run: systemctl enable proxlb
   Command #2 run: systemctl start proxlb
   e.g.:

```
root@pve101:~# systemctl enable proxlb
root@pve101:~# systemctl start proxlb
root@pve101:~#
```


8. Test ProxLB by placing one node into maintenance mode

   Command #1 run: vi /etc/proxlb/proxlb.yaml

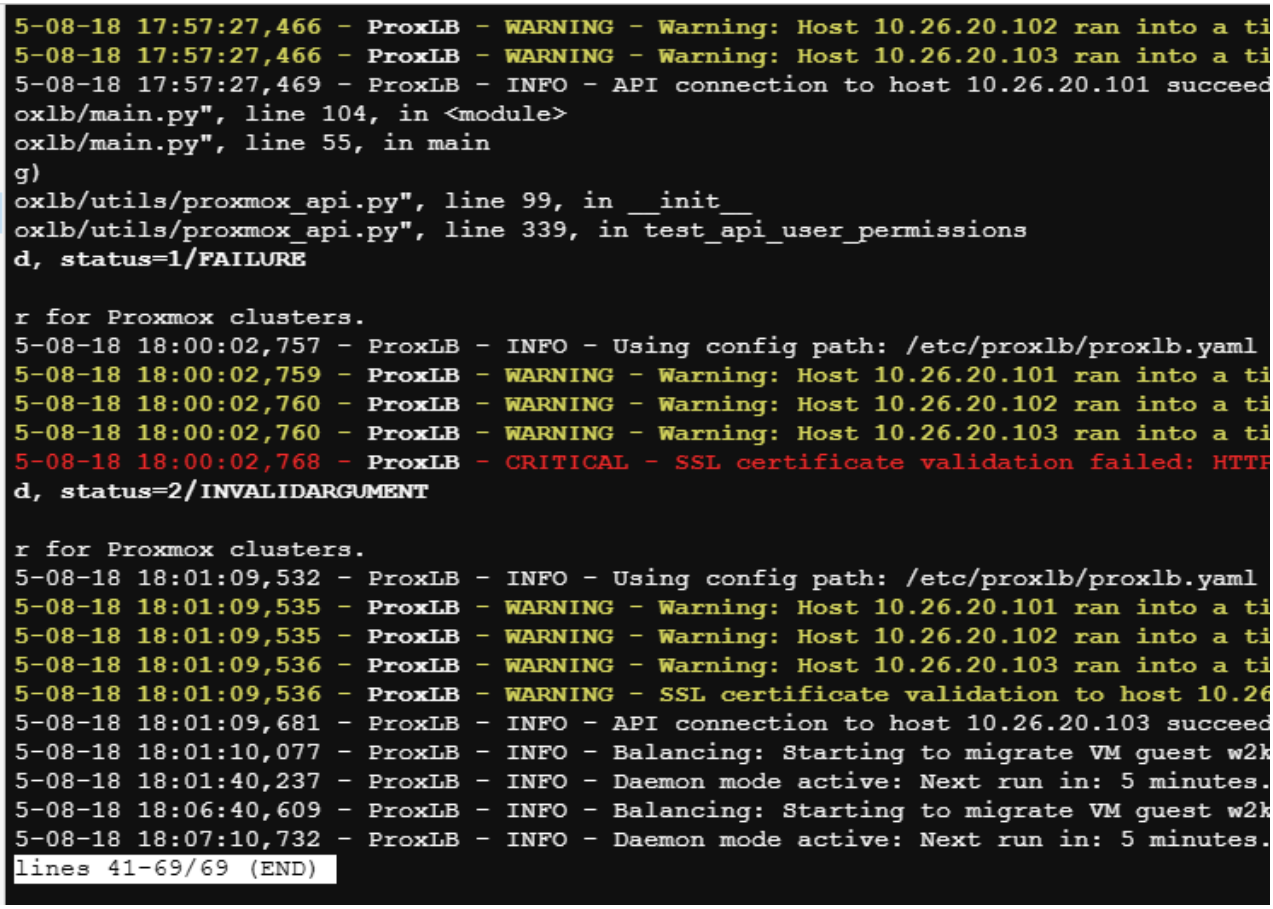
   e.g.:

```
  # API Token method
  token_id: proxlb101
  token_secret: a556da8d-78dd-42b2-8aa2-b8e8dbb9b004
  ssl_verification: True
  timeout: 10
  # API Connection retries
  retries: 1
  wait_time: 1

proxmox_cluster:
  maintenance_nodes: ['10.26.20.102']
  ignore_nodes: []
  overprovisioning: True

balancing:
  enable: True
  enforce_affinity: False
  parallel: False
  # If running parallel job, you can define
  # the amount of prallel jobs (default: 5)
  parallel_jobs: 1
  live: True
  with_local_disks: True
  balance_types: ['vm', 'ct']
  max_job_validation: 1800
  balanciness: 5
  method: memory
  mode: used

:wq
```

9. Diagnosis

Command #1 run: journalctl --grep "proxlb”

e.g.:



NOTE : You will see warnings and failures due to ProxLB starting before configuration. What you are looking for is ‘Daemon mode active…’ on the last line

Software Defined Networking (SDN)

Software Defined Networking (SDN) allows for abstraction of the network settings defined on each node across the cluster. It further allows for the defining of specific user permissions to specific Zones, allowing access only to the network(s) defined in that Zone.

Zone Types

Simple:

A Simple Zone exists across the cluster, however restricts networking to a single node.

VLAN:

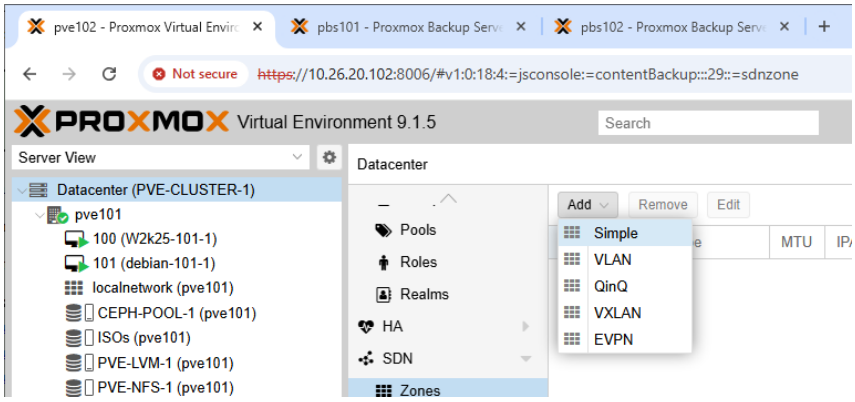
A VLAN Zone allows for the assignment of network resources and VLAN tagsv101

SBS LAB – (GUI) SDN Simple Zone

1. Create a Simple Zone

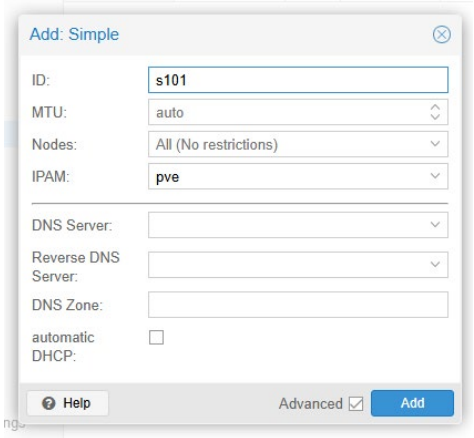
Step 1: Datacenter > SDN > Zones > Add > Simple

e.g.:



Step 2: Name the Zone: sXYZ

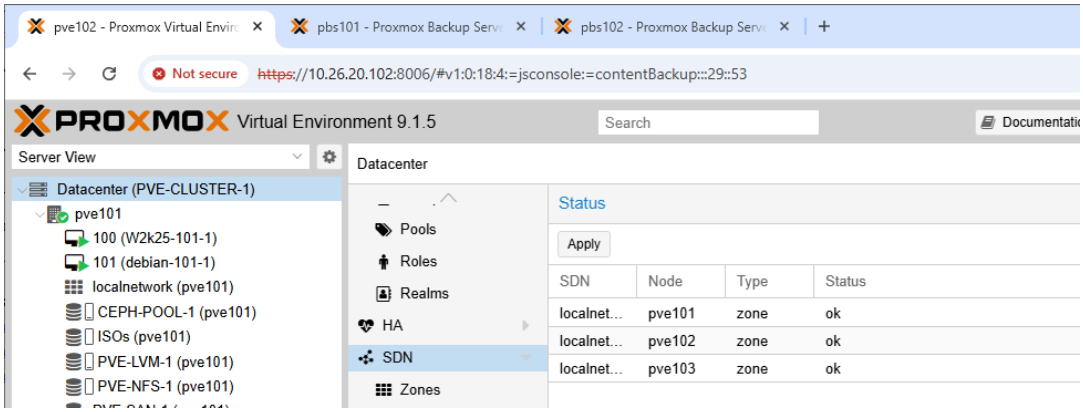
e.g.:



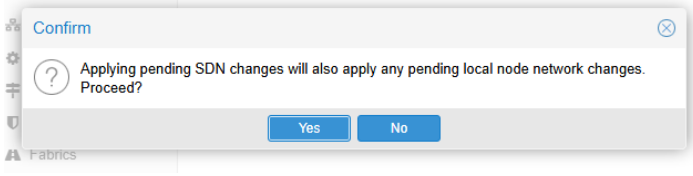
NOTE : The Zone name is limited to 8 characters, no special characters

Step 3: Datacenter > SDN > Apply

e.g.:

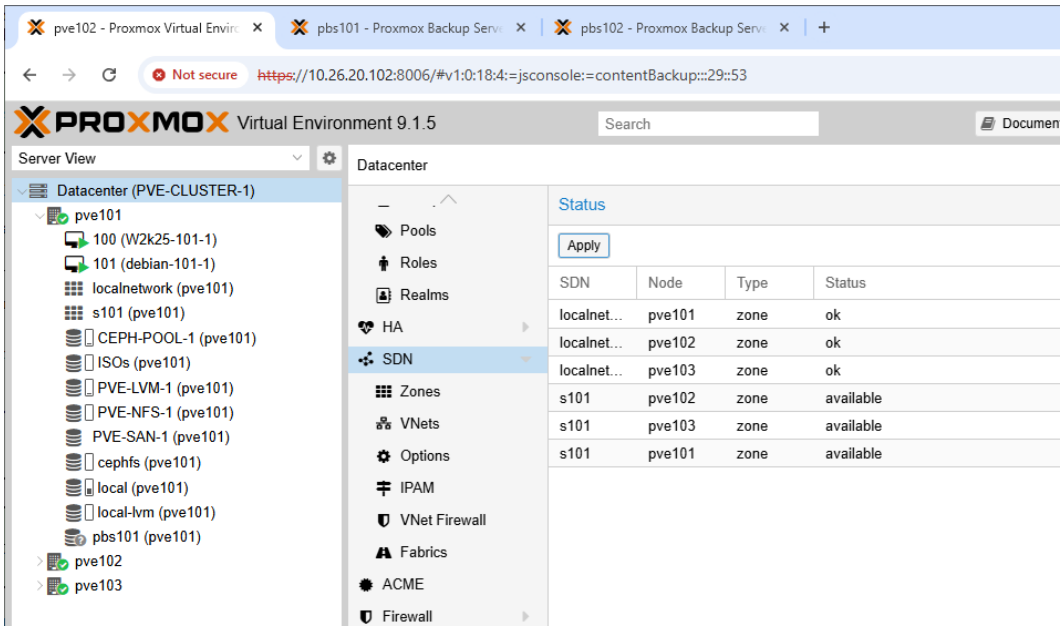


e.g.:



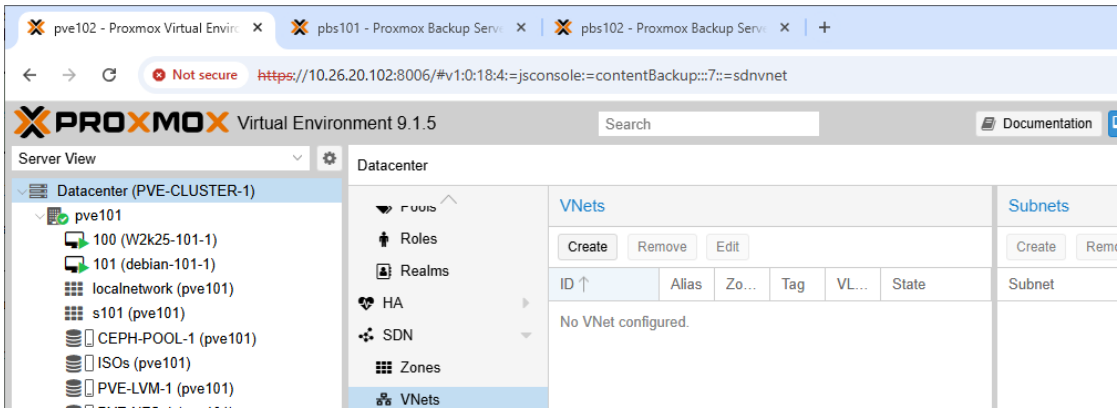
NOTE : Wait 30-45 seconds

e.g.:



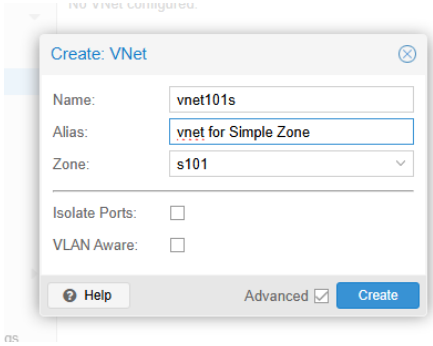
Step 4: Datacenter > SDN > VNets > Create

e.g.:



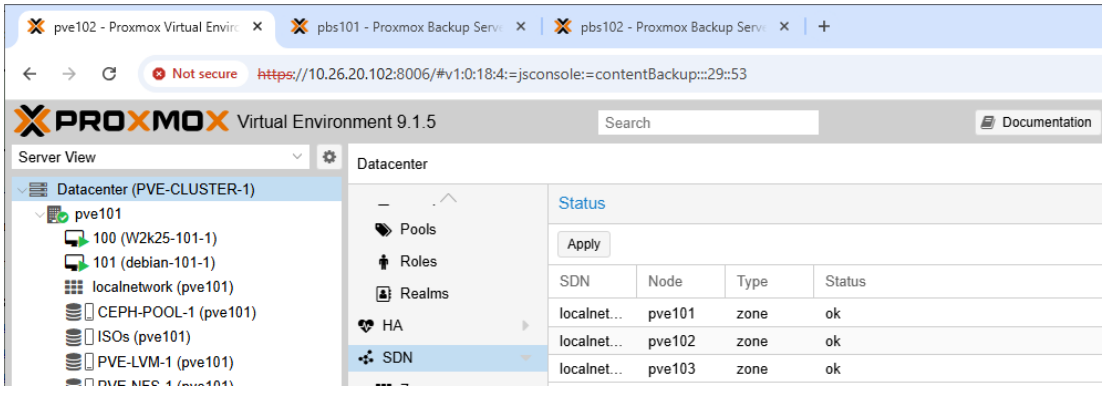
Step 5: Create: vnetXYZ in Zone: sXYZ

e.g.:



Step 6: Datacenter > SDN > Apply

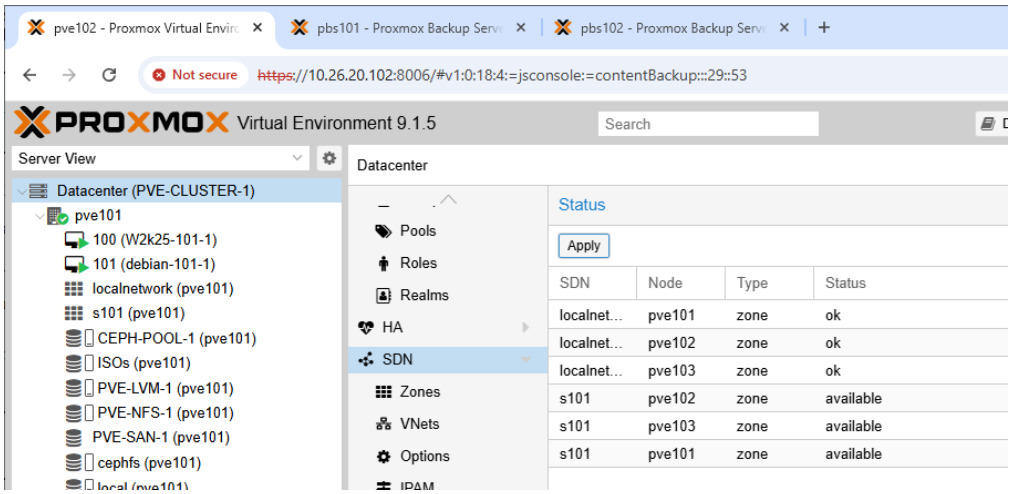
e.g.:



e.g.:

NOTE : Wait 30-45 seconds

e.g.:

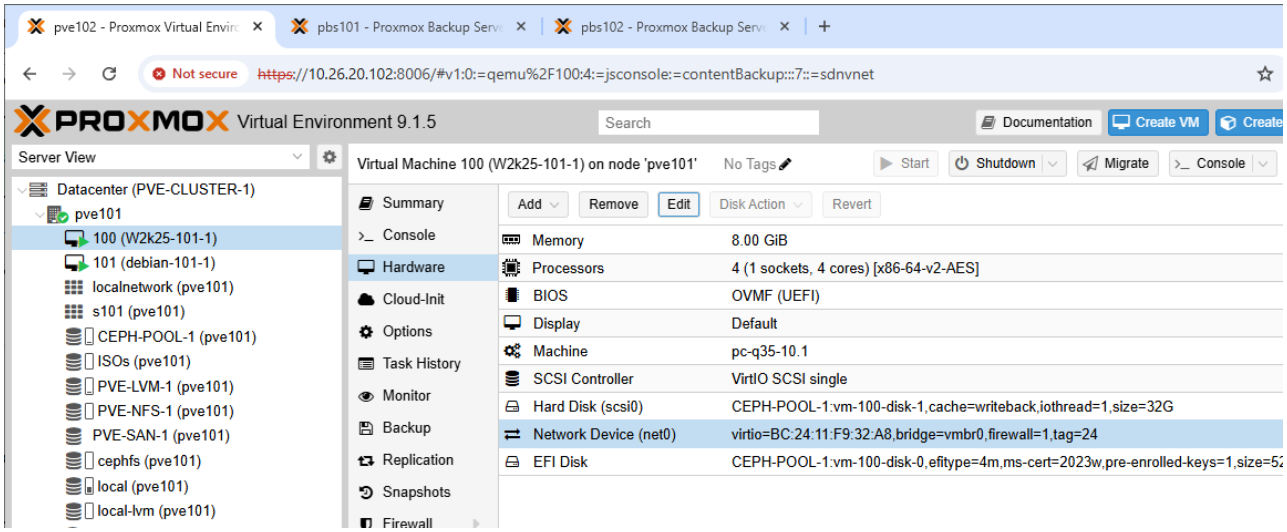


NOTE : VNets do not appear in the top-level SDN page

2. Test Simple Zone

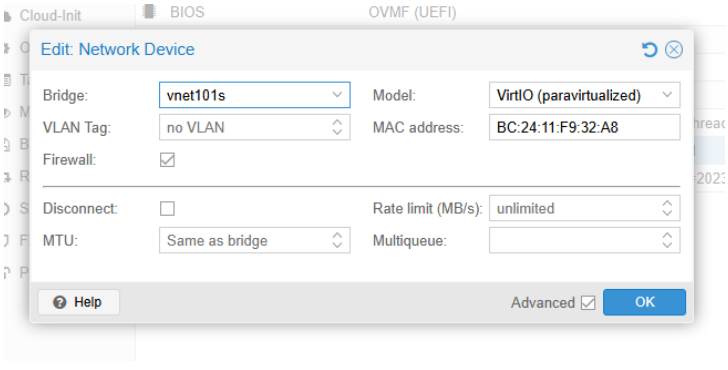
Step 1: W2k25-XYZ-1 > Hardware > Network Device > Edit

e.g.:



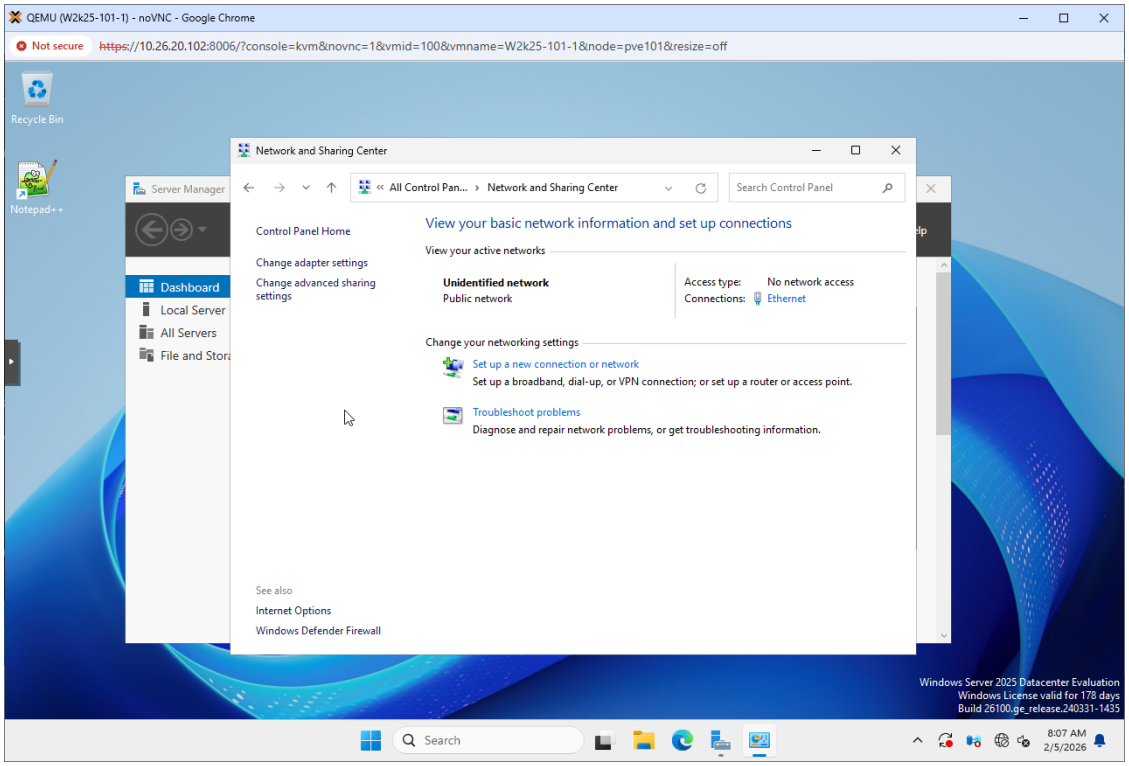
Step 2: Set the Bridge to vnetXYZs and remove the VLAN tag

e.g.:



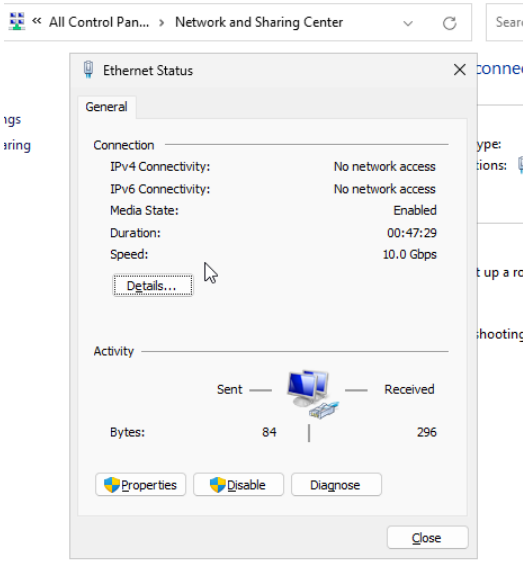
Step 3: On your W2k25-XYZ-1 VM, open Control Panel > Network and Sharing Center > Double-click your adapter

e.g.:



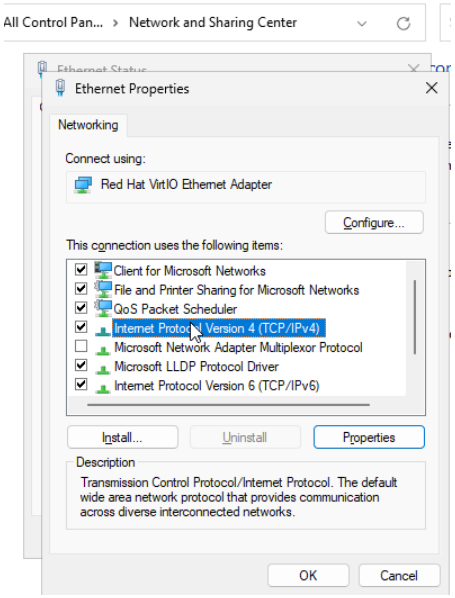
Step 4: Properties

e.g.:



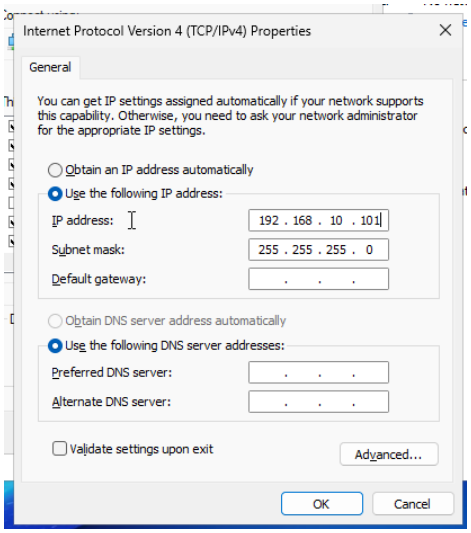
Step 5: Internet Protocol Version 4 (TCP/IPv4) > Properties

e.g.:



Step 6: Set the IP to: 192.168.10.XYZ

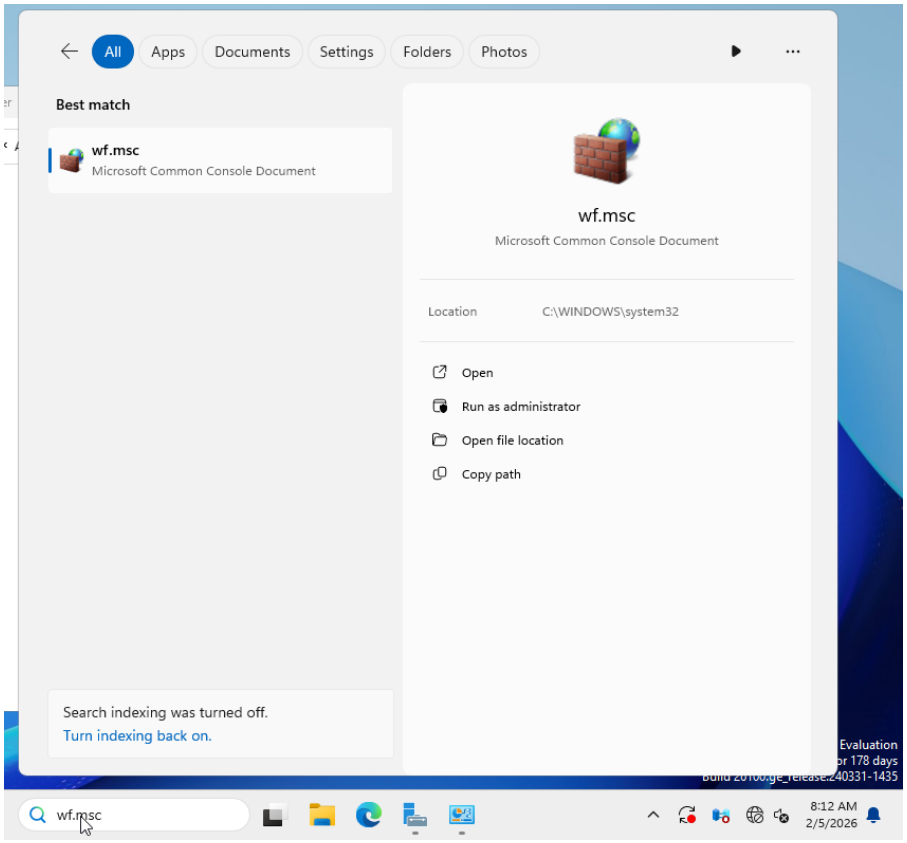
e.g.:



NOTE : Be sure to OK and Close

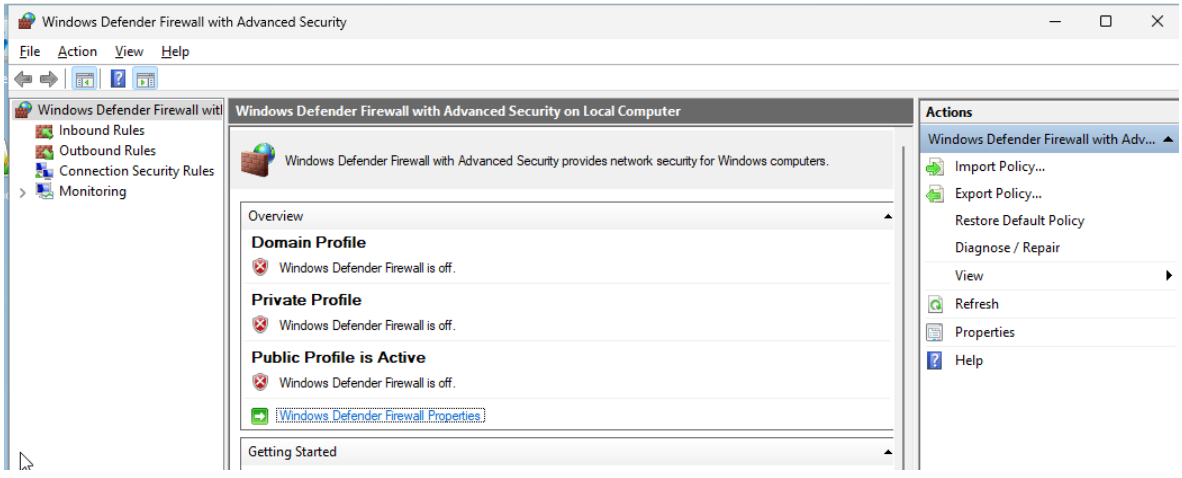
Step 7: Open Windows Firewall by typing wf.msc in search

e.g.:



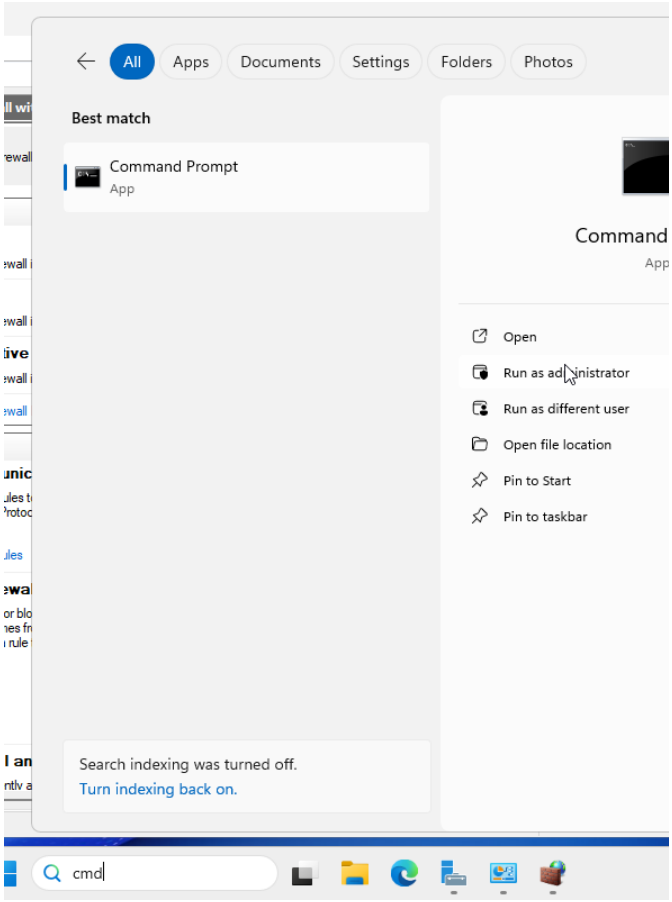
Step 8: Disable Windows Firewall completely

e.g.:



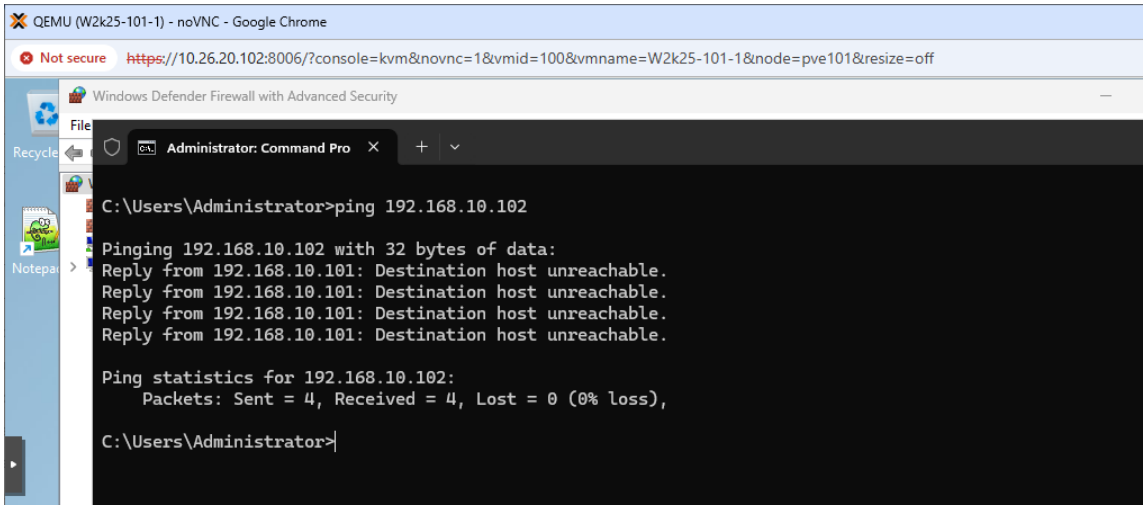
Step 9: Open a CMD prompt

e.g.:



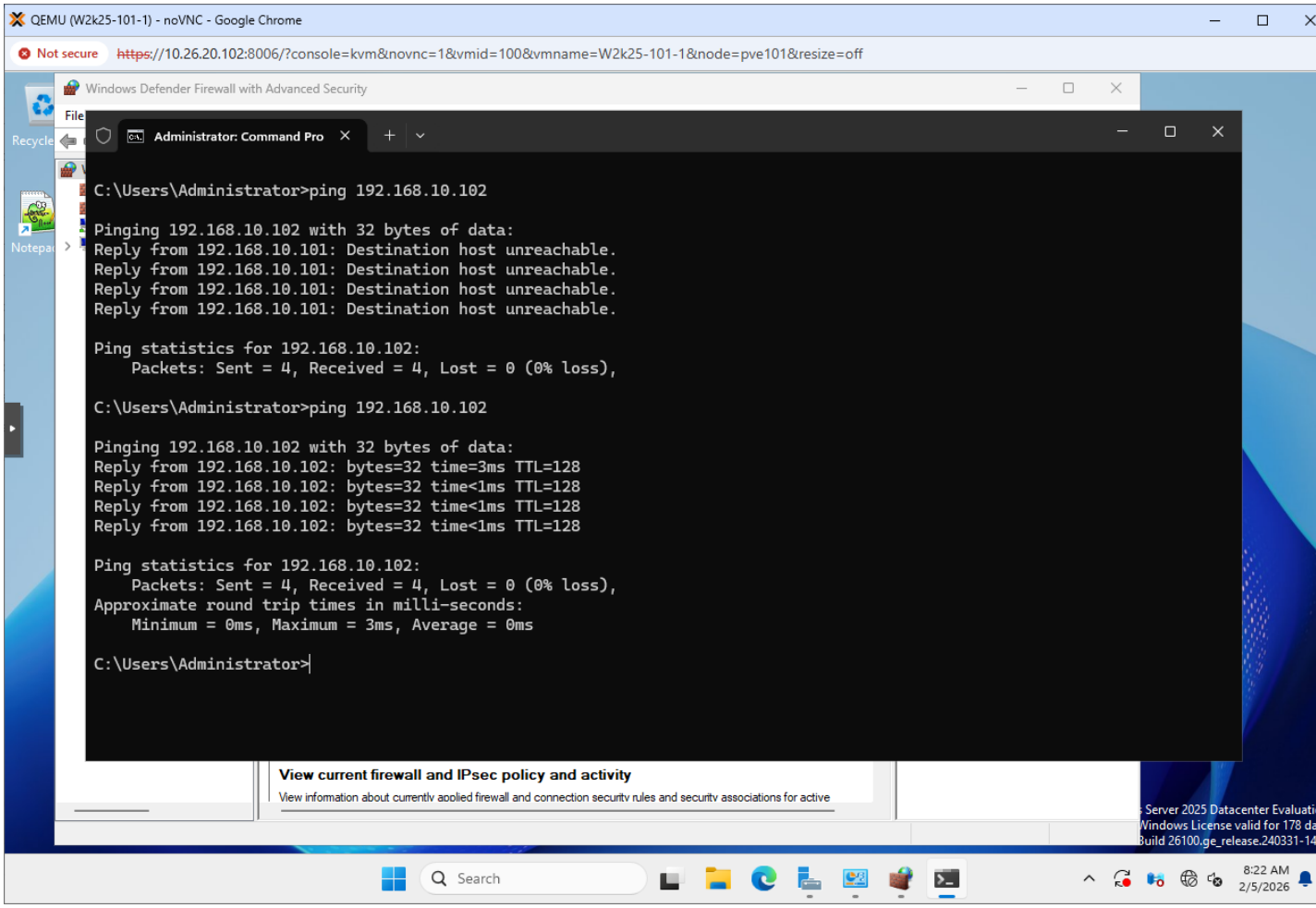
Step 10: Ping another VM on the same VNet

e.g.:



Step 11: If the ping WAS successful, your VM was on the same node as the VM you were pinging, if the ping was NOT successful, migrate to the same node as the VM you are pinging

e.g.:



IN PRODUCTION:

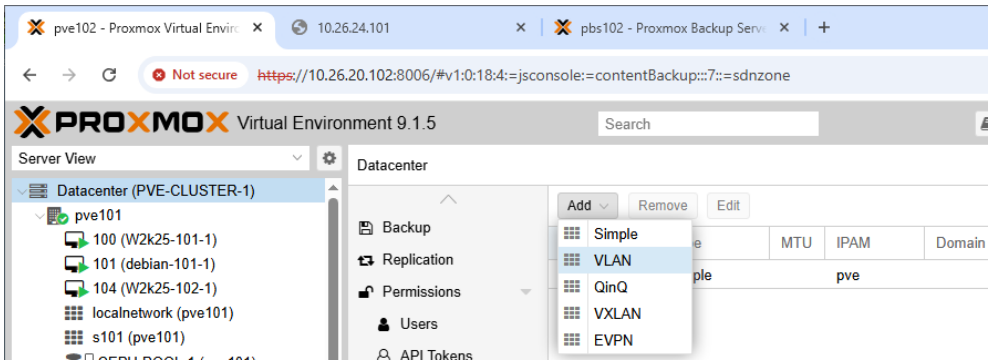
When we created the Simple Zone, we did not (could not) assign any network resources, so communication is limited to node-only. While this might be useful for single-node environments and other limited testing, Simple Zones are not very useful overall.

SBS LAB – (GUI) SDN VLAN Zone

1. Create a VLAN Zone

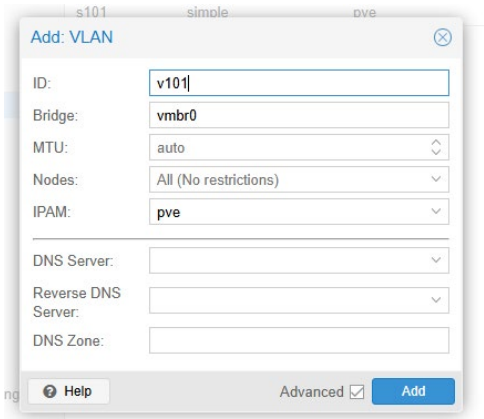
Step 1: Datacenter > SDN > Zones > Add > VLAN

e.g.:



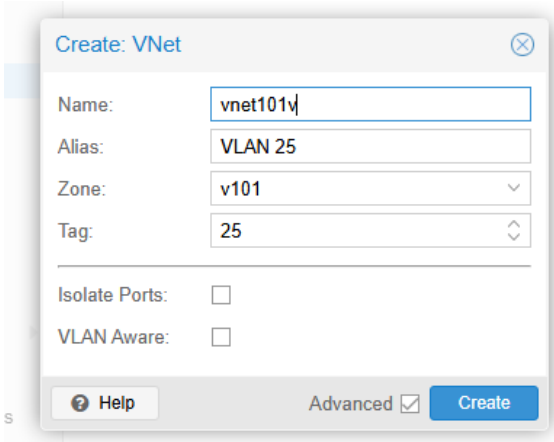
Step 2: ID will be vXYZ and bridge will be vmbr0

e.g.:



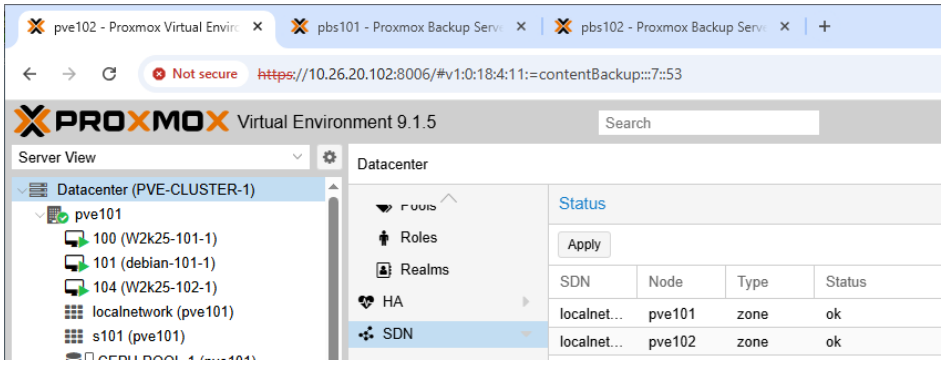
Step 3: Create a VNet on your VLAN Zone, specifying VLAN 25

e.g.:

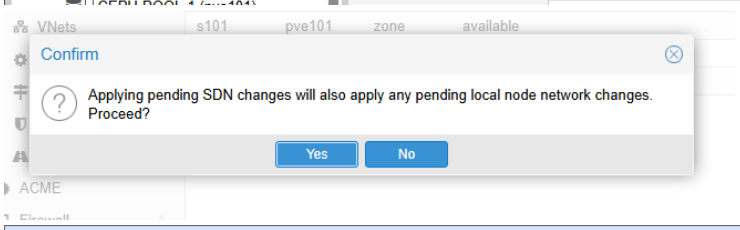


Step 4: Datacenter > SDN > APPLY

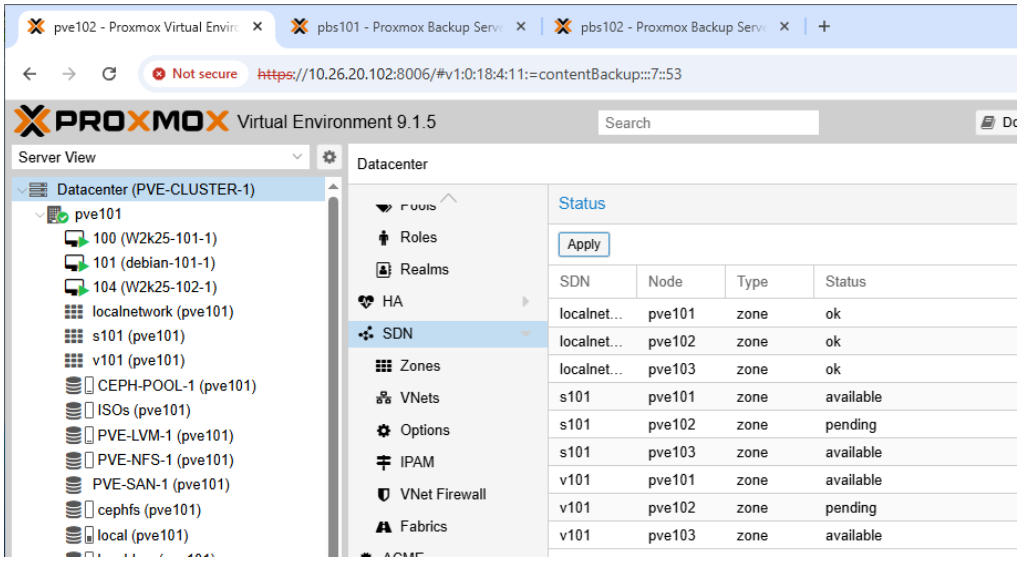
e.g.:



e.g.:

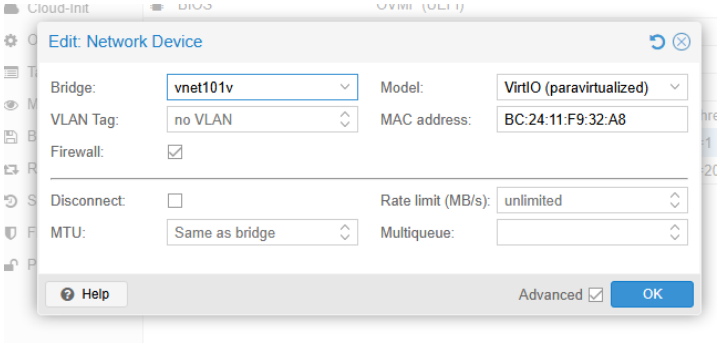


e.g.:



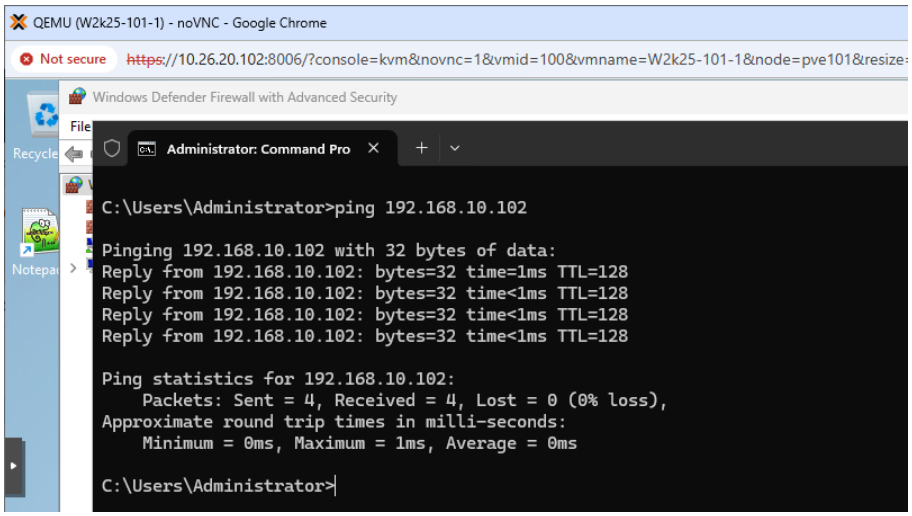
Step 5: W2K25-XYZ-1 > Hardware > Network Device > Edit

e.g.:



Step 6: Now try to ping another W2K25 VM on the same vnet

e.g.:

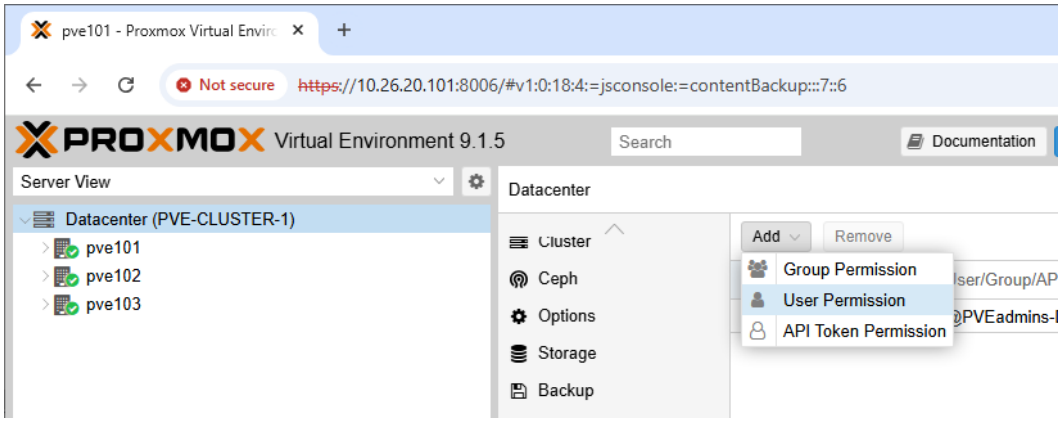


IN PRODUCTION: This will work (if we got it right) on the same node or across nodes. The VLAN Zone and corresponding VNet is much more useful as we assign network resources to it.

SBS LAB – (GUI) Permissions for SDN

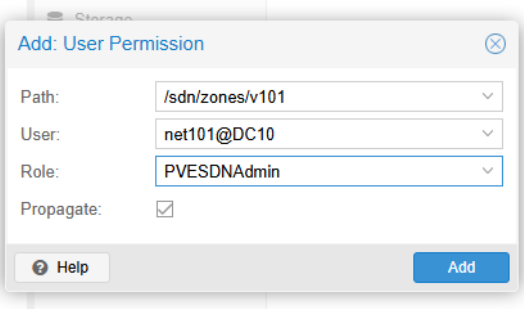
Step 1: Datacenter > Permissions > Add > User Permission

e.g.:



Step 2: Under Path, select /sdn/zones and then **manually type the name of your VLAN Zone: vXYZ**, select the user netXYZ and the Role PVESDNAdmin

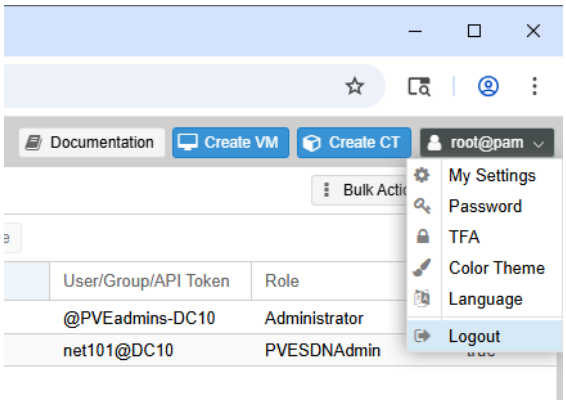
e.g.:



NOTE : Your VLAN Zone is ***supposed*** to appear in the dropdown, however at this writing, some glitch in the interface means you have to type the zone in manually.

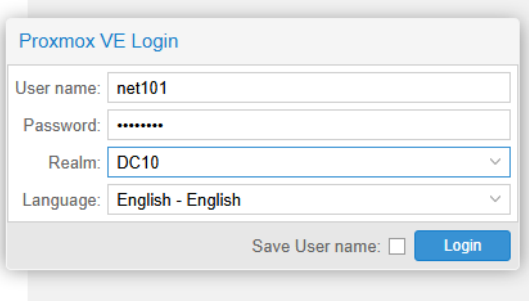
Step 3: Log out

e.g.:

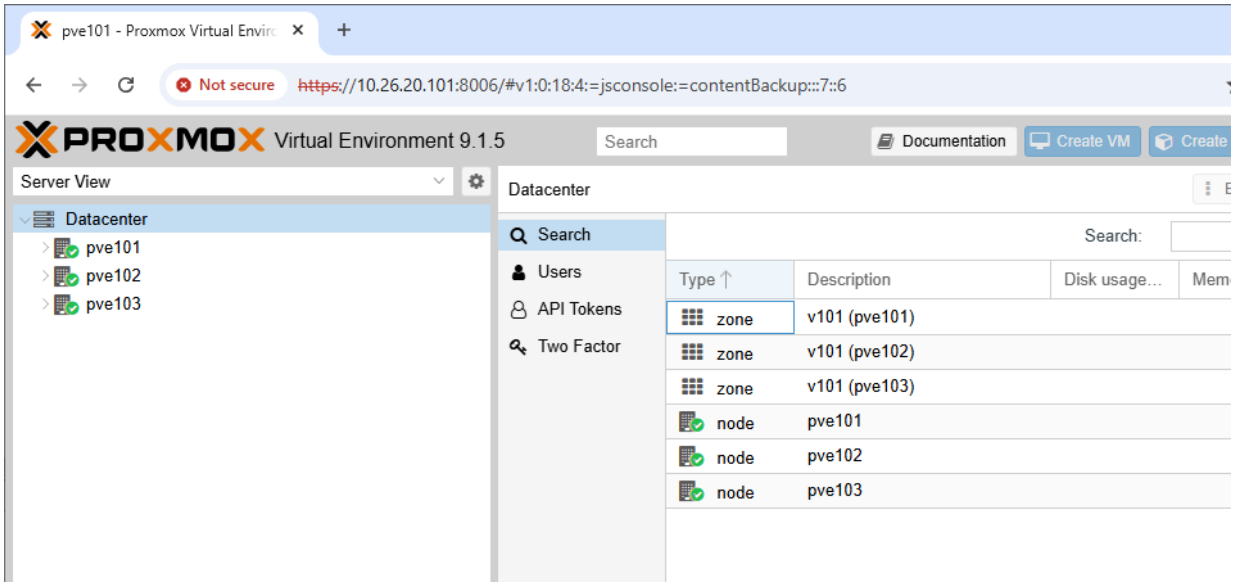


Step 4: Log back in as netXYZ

e.g.:



e.g.:



Updates and Upgrades

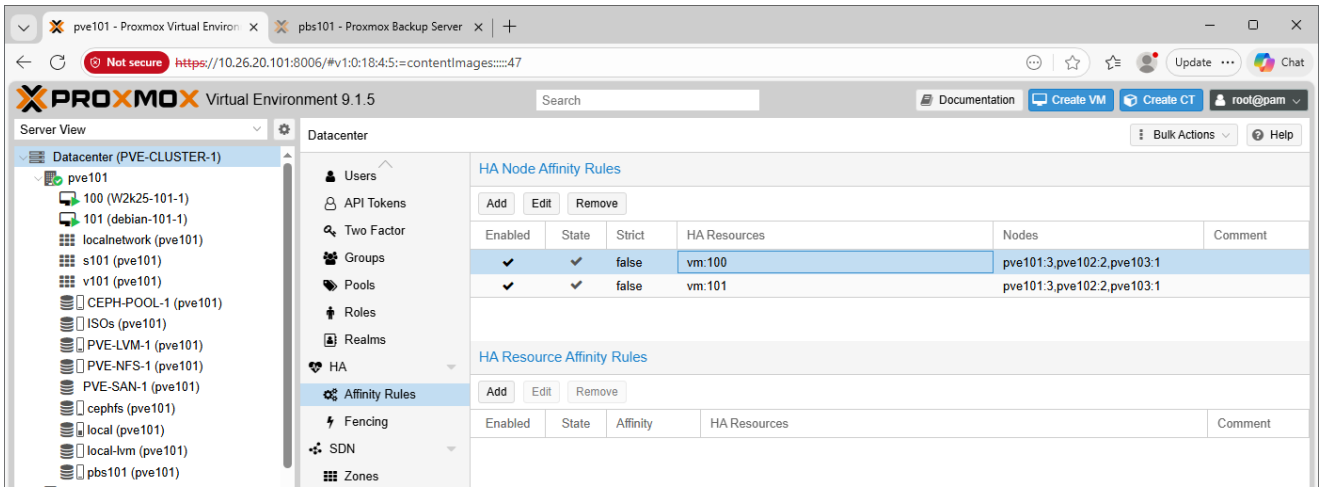
SBS LAB – (GUI/CLI) Upgrading to Proxmox 9.2

1. Disable HA Affinity to a particular node for all VMs which have HA Affinity set

   Step 1: Datacenter > HA > Affinity Rules

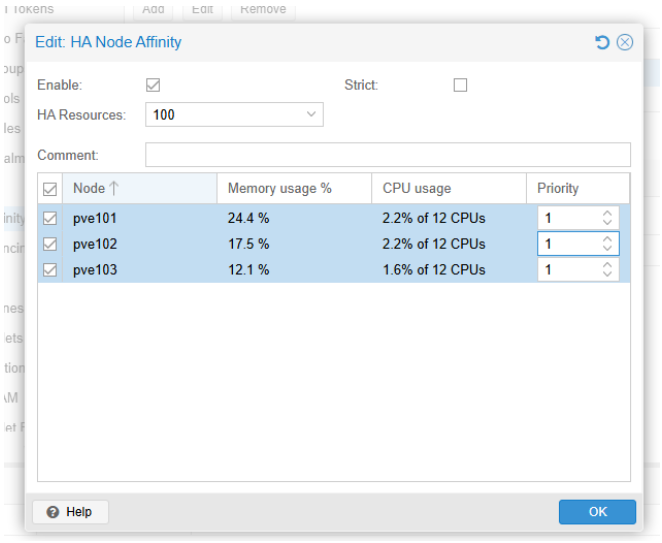
   Step 2: Select VM > Edit

   e.g.:



   Step 3: Set all HA Affinity for the VM to the same priority

   e.g.:

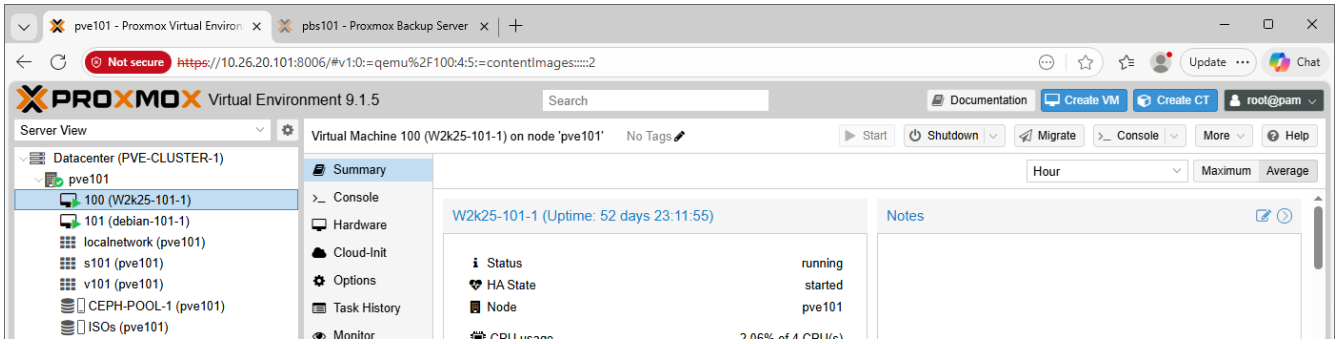


   Step 4: Repeat for all your VMs (name-XYZ-X)

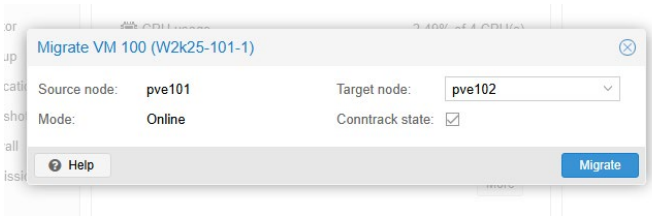
2. Migrate all of your active workloads off the PVE node to be upgraded

   Step 1: Select a VM > Migrate

   e.g.:



   e.g.:



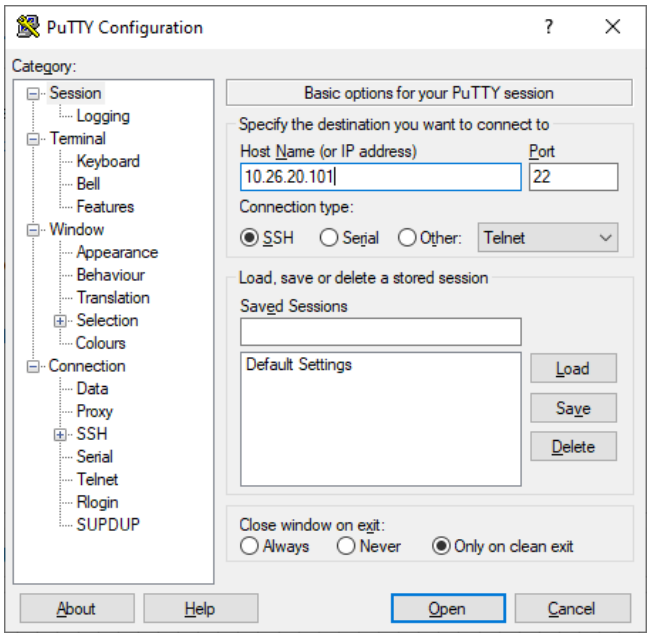
   e.g.:

   NOTE : If VM has HA Affinity to a particular node, HA affinity must be disabled or the VM will migrate right back to the source node

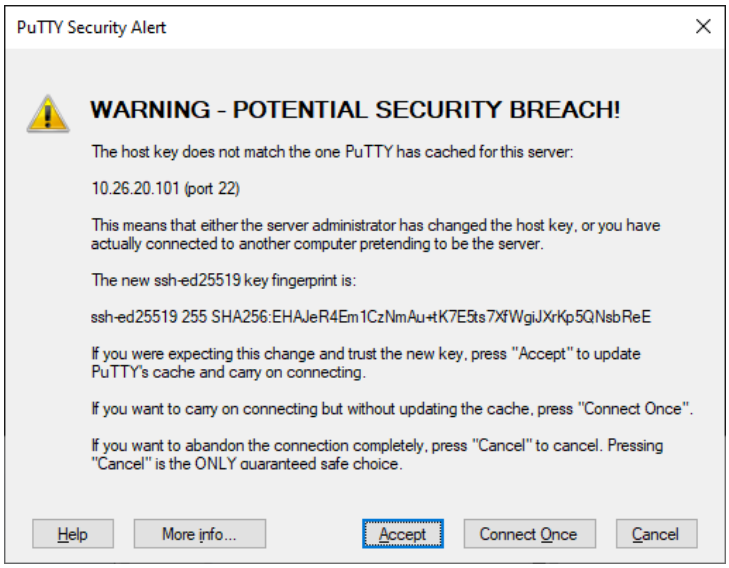
1. Method 1 (CLI) Preferred

Step 1: Connect to PVE node with SSH client (Putty)

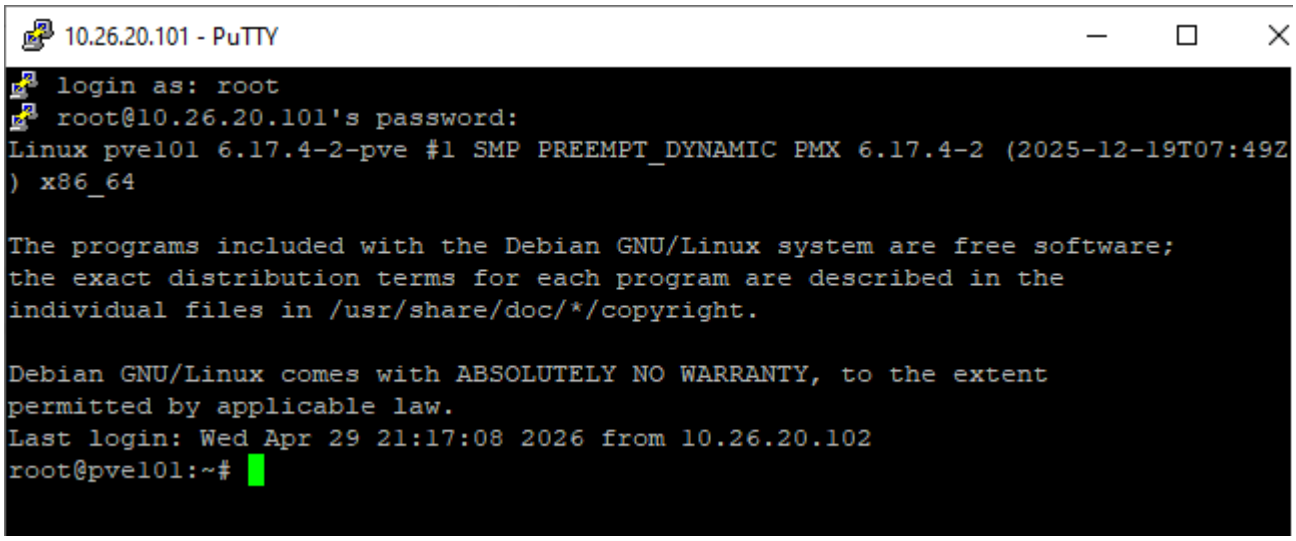
e.g.:



e.g.:



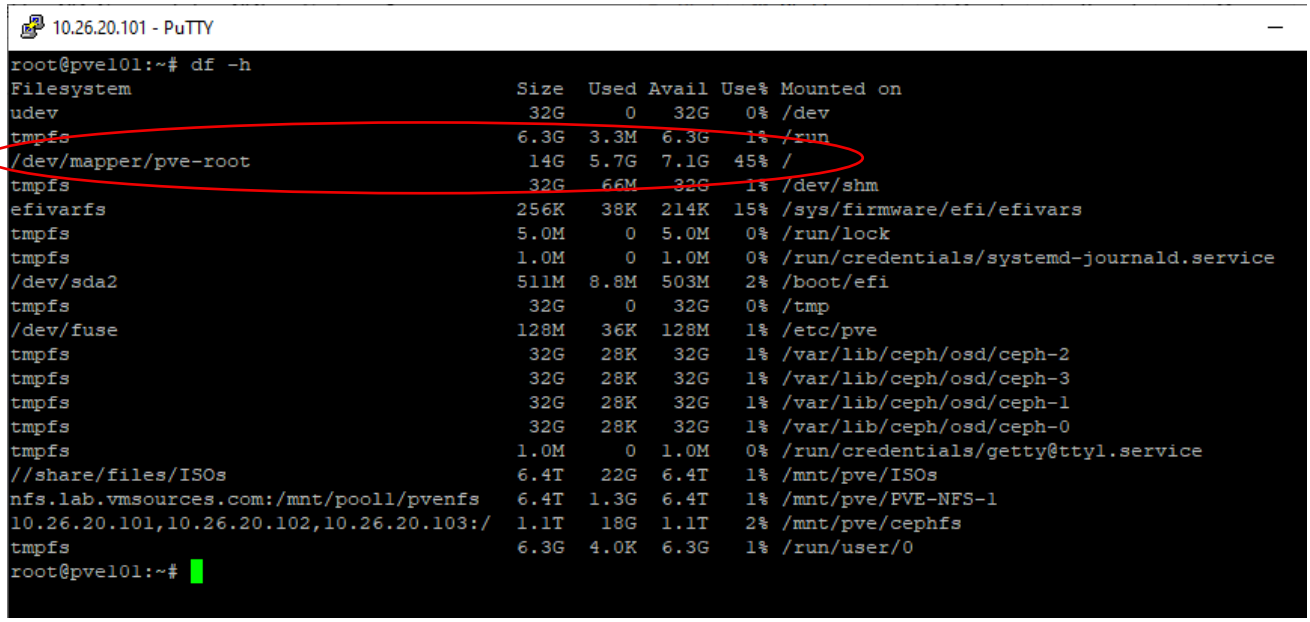
e.g.:



Step 2: Make sure you have at least 5GB free on root filesystem /dev/mapper/pve-root

Command #1 run: df -h

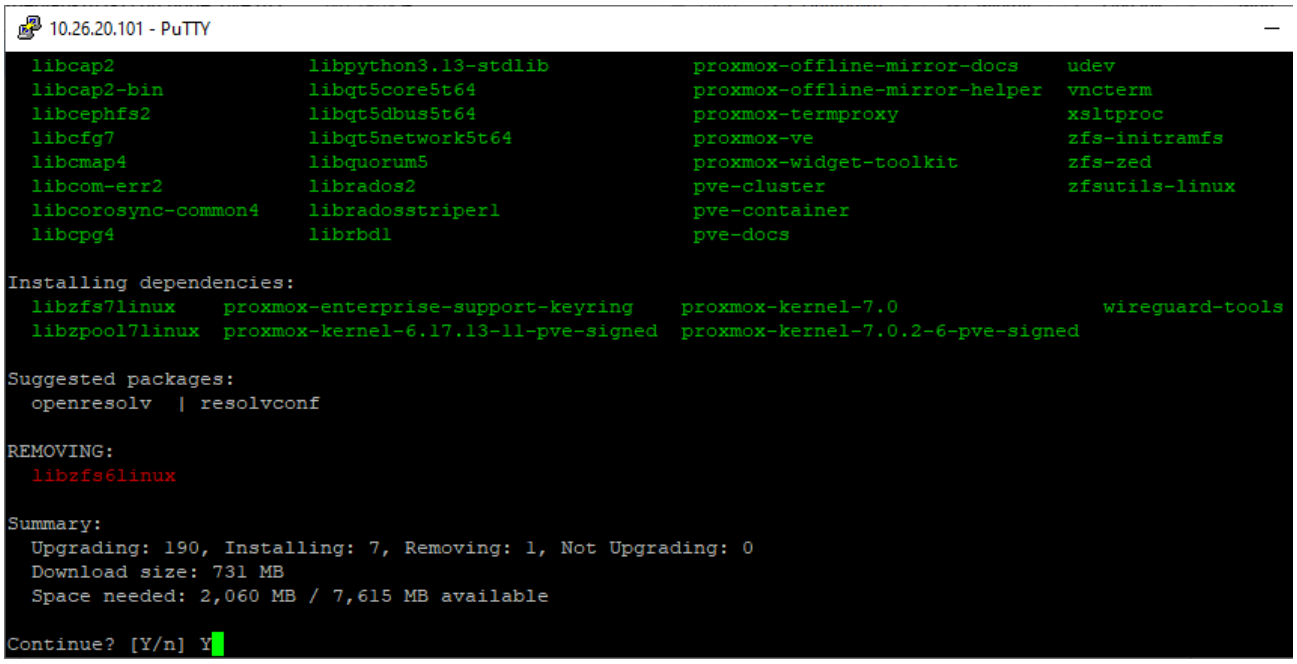
e.g.:



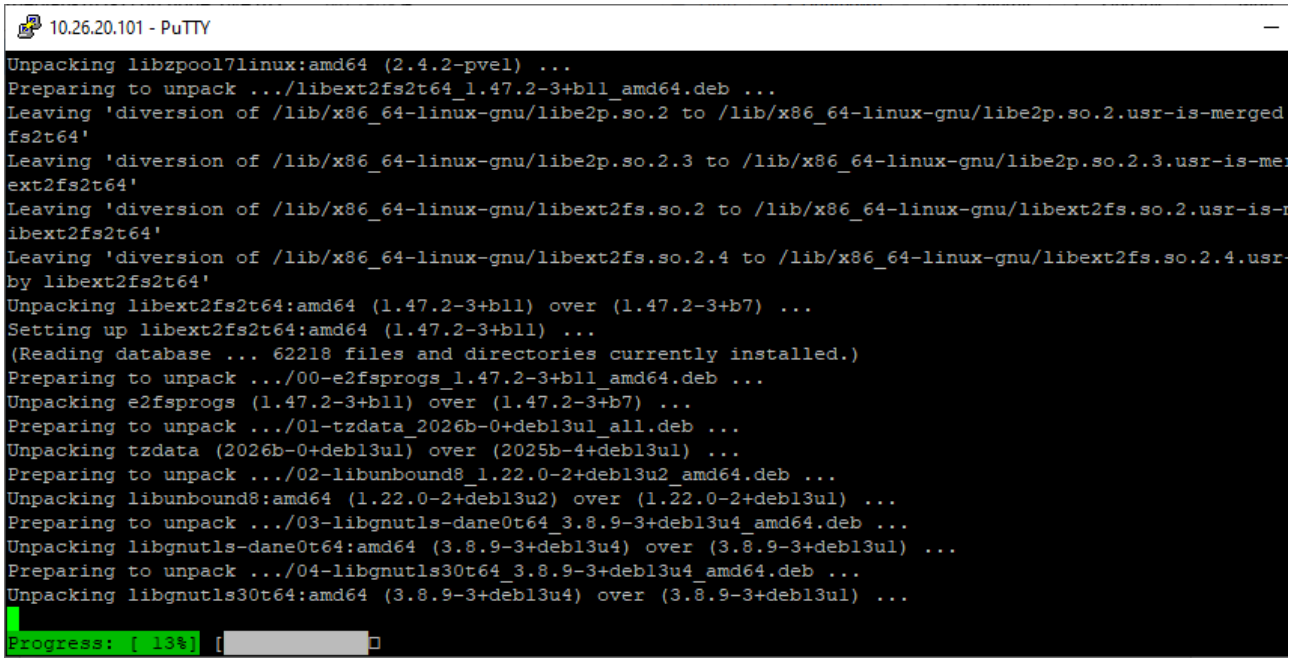
Step 3: Upgrade PVE

Command #1 run: apt update && apt dist-upgrade

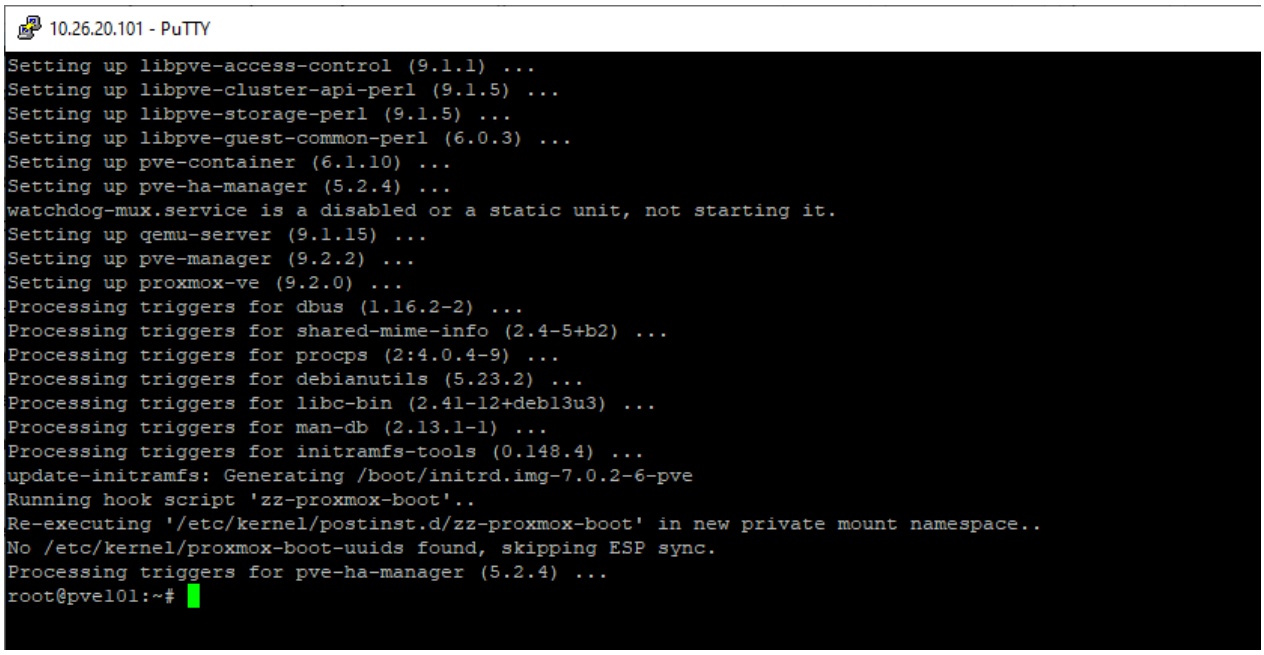
e.g.:



e.g.:



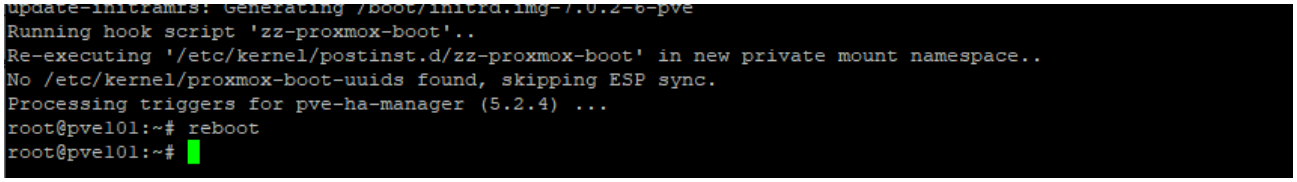
e.g.:



Step 4: Reboot

Command #1 run: Reboot

e.g.:

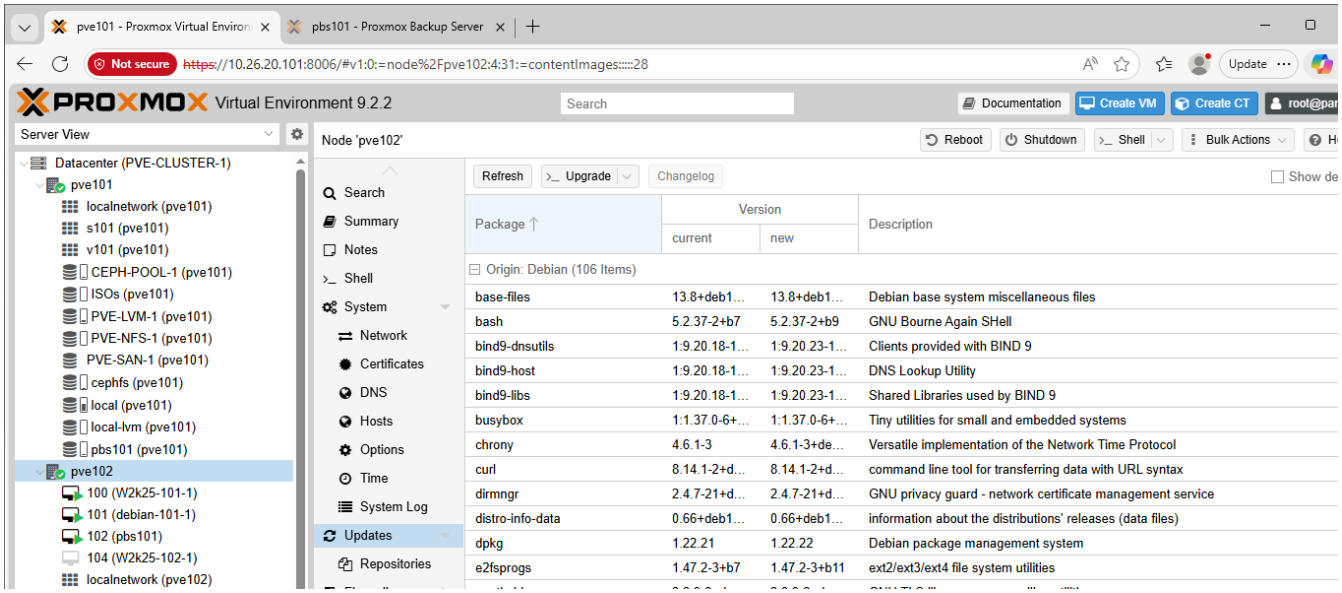


NOTE : If you were logged in (GUI) to the node that was rebooted, you can connect to another cluster node to watch and see when it is back online, or just wait

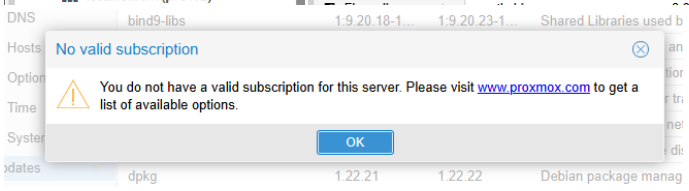
2. Method 2 (GUI/CLI)

Step 1: Select node pveXYZ > Updates > Refresh

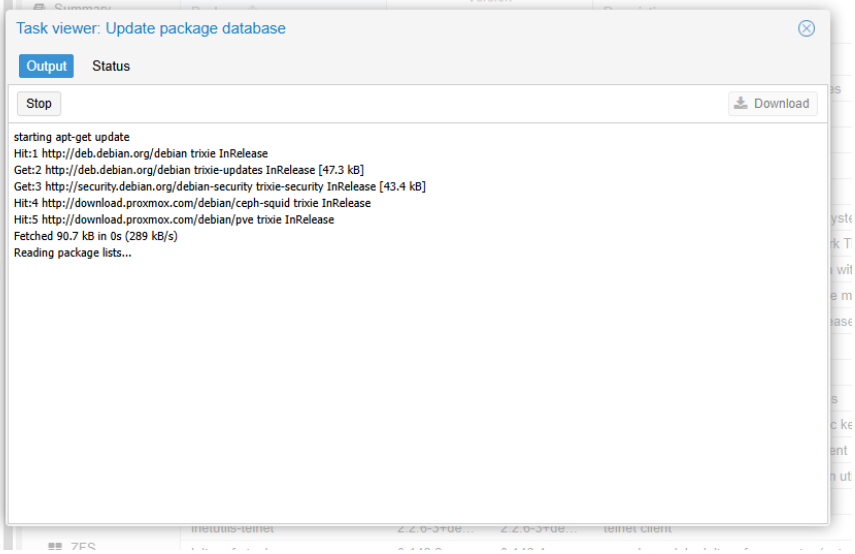
e.g.:



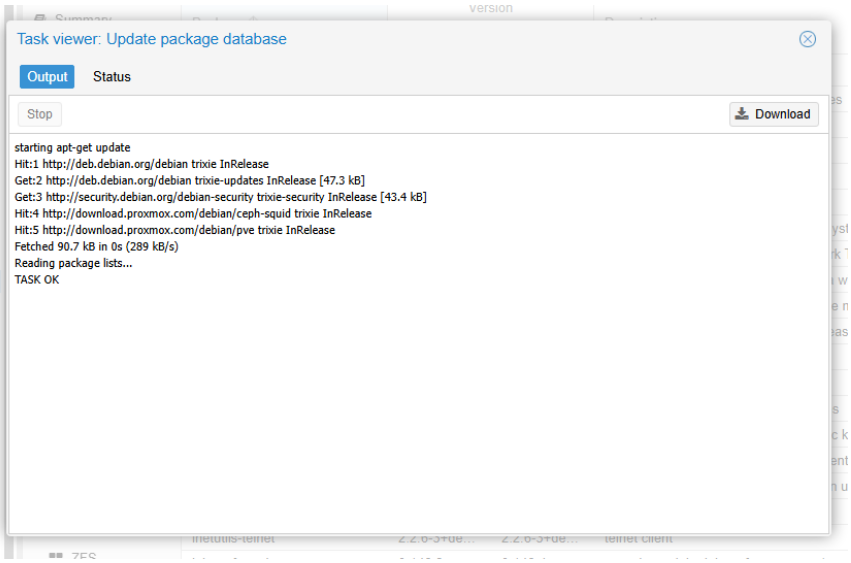
e.g.:



e.g.:



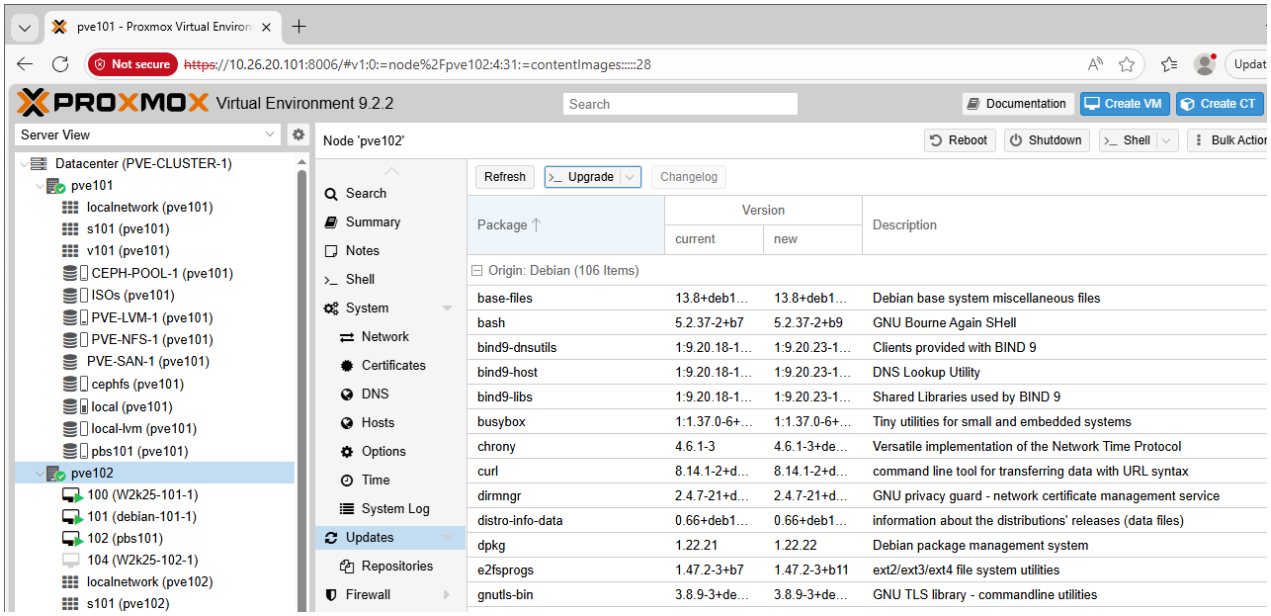
e.g.:



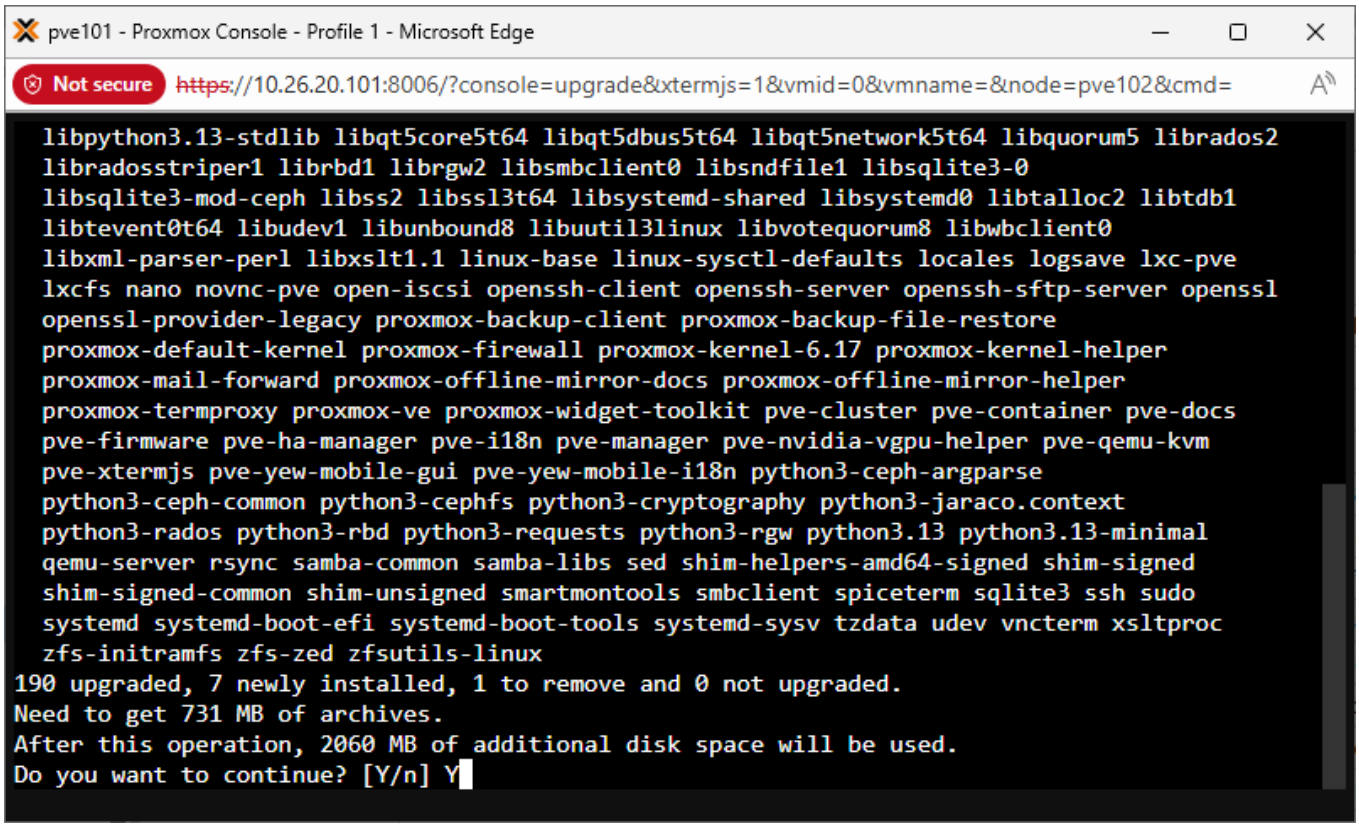
e.g.:

Step 2: Upgrade

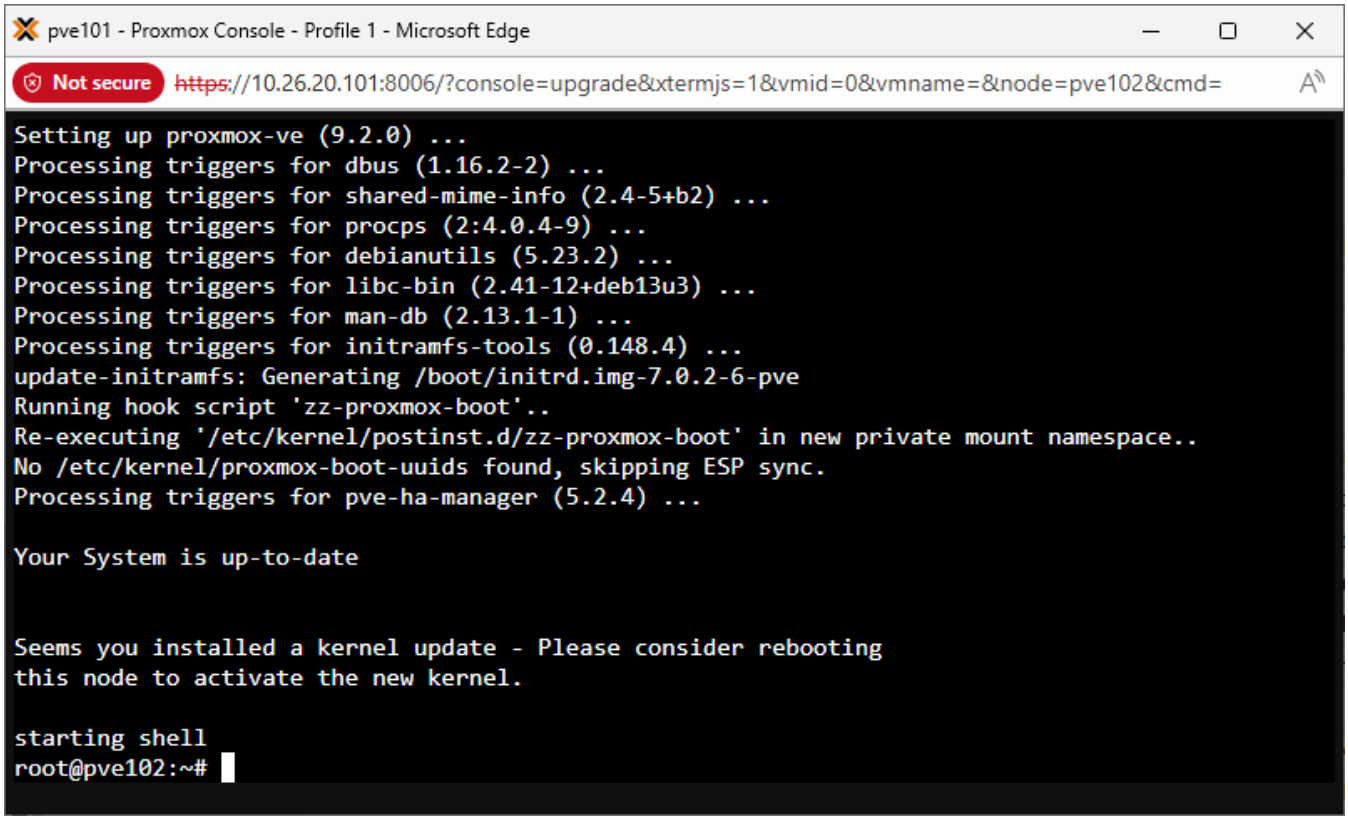
e.g.:



e.g.:



e.g.:



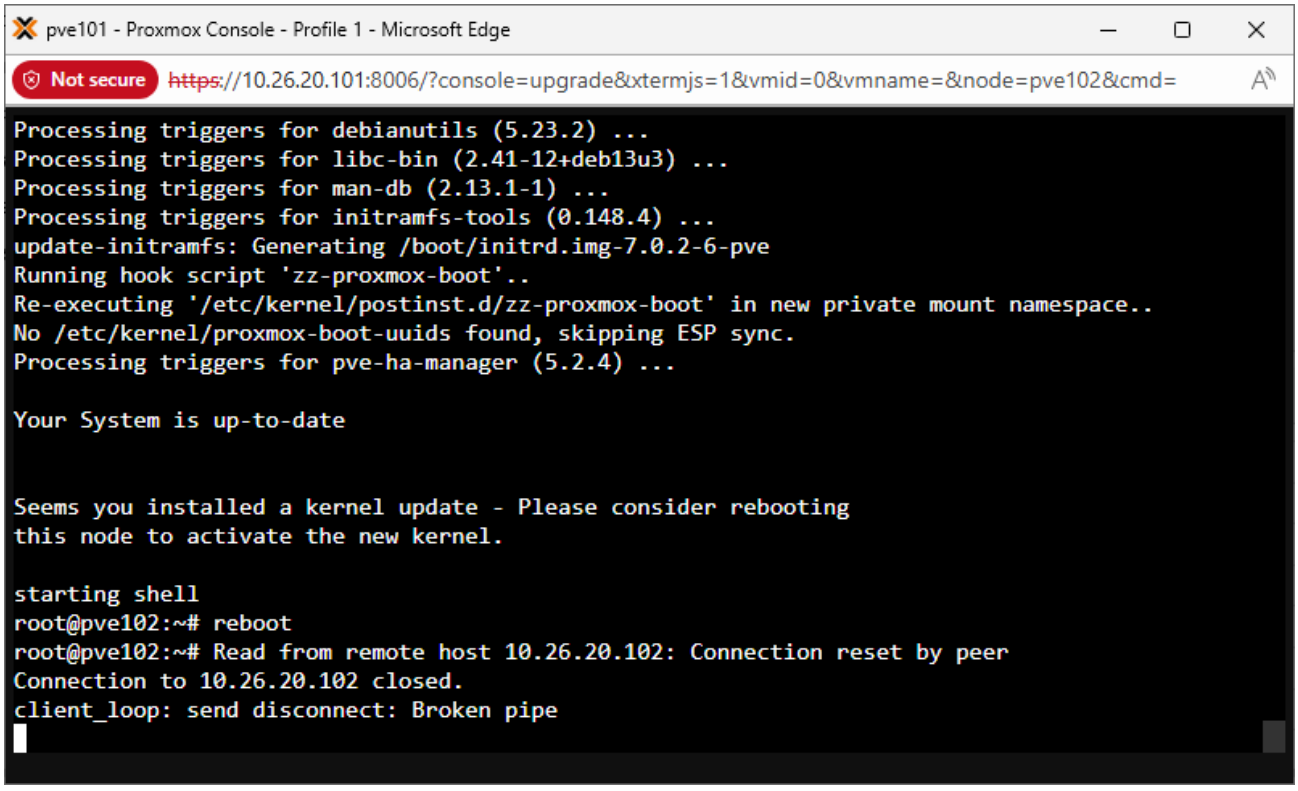
NOTE : You might notice that I failed to migrate my VMs off of this node before upgrading. This actually highlights the rolling update/upgrade capability of Debian and APT, however since

a new kernel was installed and a reboot required, we will need to migrate VMs before the reboot!

Step 3: Reboot

Command #1 run: reboot

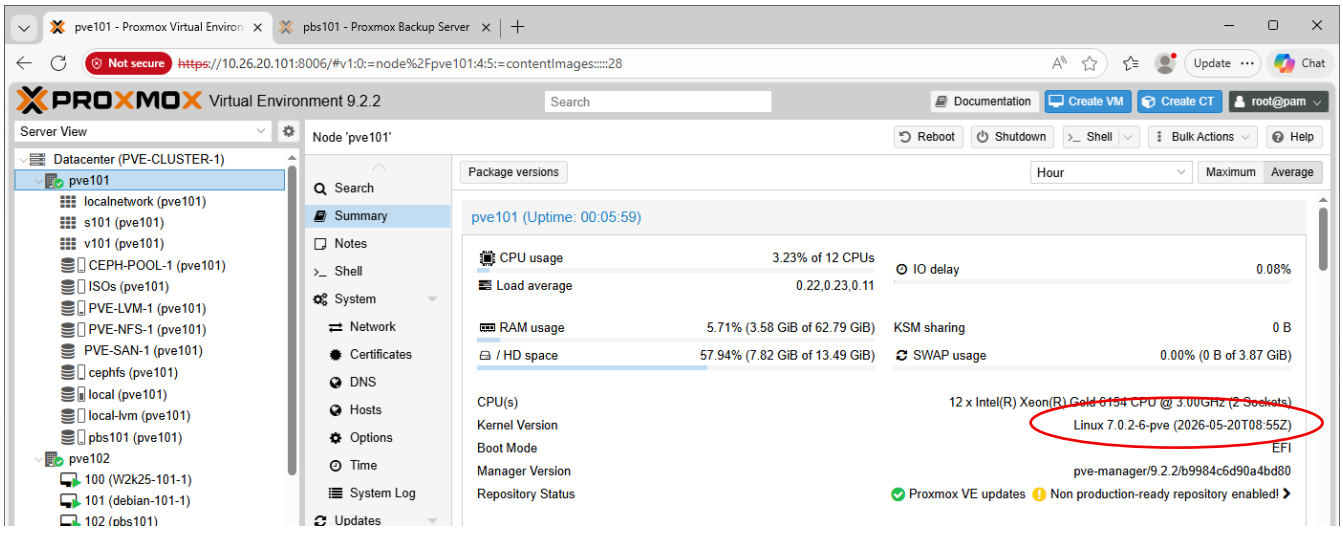
e.g.:



3. Verify the process

Step 1: Verify upgrade status

e.g.:



Step 2: Proceed with remaining nodes using your preferred method

Backups and Disaster Recovery

Backups with Proxmox Backup Server (PBS)

Proxmox Backup Server (PBS) is a platform-integrated backup and recovery utility that installs as a Server and a Client. PBS differs from most Enterprise backup solutions in that it is push-based and backups are configured at the source, rather than on the PBS server.

PBS is, however, capable of implementing replication to another PBS on either a pull or push based schedule for DR purposes.

PBS has a relatively limited storage (repository) capability. It can use:

- Disks or LUNs (mounted to PBS)
- Removable disks (mounted to PBS)
- S3 as a Technology Preview

PBS functions by indexing protected Workloads (VMs & Containers) and then protecting the data by saving it in “Chunks,” each Chunk identified by a hash which is indexed on PBS. From a storage perspective, this is very efficient, because even dissimilar Workloads will have identical bits of data. If a Chunk exists and PBS determines that the hash matches, that Chunk is not duplicated, merely indexed for that workload.

Backup Storage for PBS

Many would argue that PBS is best installed on a physical server with a large disk capacity. While installing PBS as a physical server is great, it defeats the entire purpose for which we are running Proxmox in the first place!

We prefer installing PBS as a VM running on PVE and connecting to remote storage.

Because of the way PBS saves data, NFS and SMB are a particularly poor choice as a repository subsystem due to the large number of files (although technically usable). PBS functions best when connected to block storage.

Affordable iSCSI devices (like Buffalo, Synology) provide the ability to configure an iSCSI target and are usually capable of RAID, making them a particularly good choice for PBS backup storage. You will want to make sure of the following factors in considering and configuring iSCSI storage for PBS:

- Isolated iSCSI networks (no gateway)
- Strong passwords for the iSCSI device and PBS
- Consider using CHAP and ACL’s on the PBS and iSCSI device
- Choose RAID6 when possible to protect against device failure

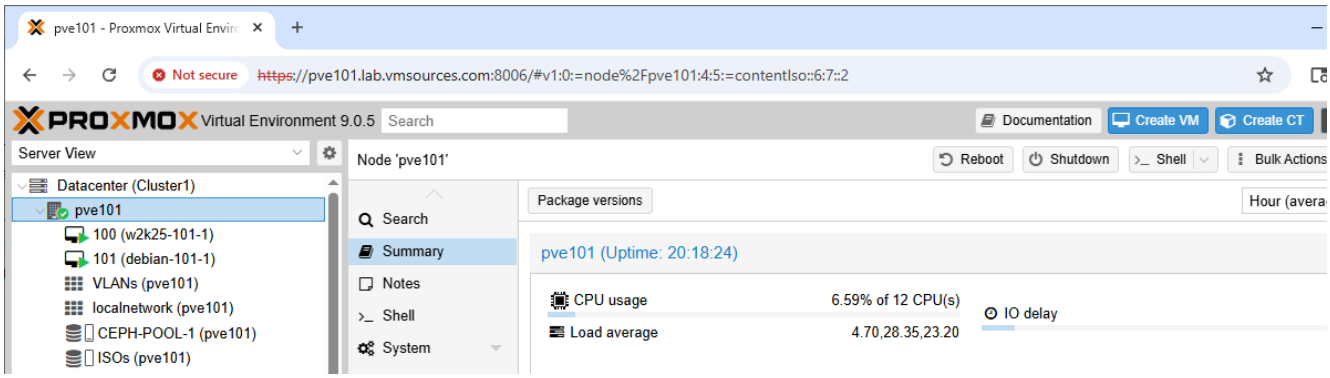
- Choose an iSCSI device that has at least two NICs, so one can be configured for management and the other as the target.

SBS LAB – (GUI) Installing Proxmox Backup Server (PBS)

1. Create PBS as a VM

Step 1: pveXYZ > Create VM

e.g.:



Step 2: Node: pveXYZ

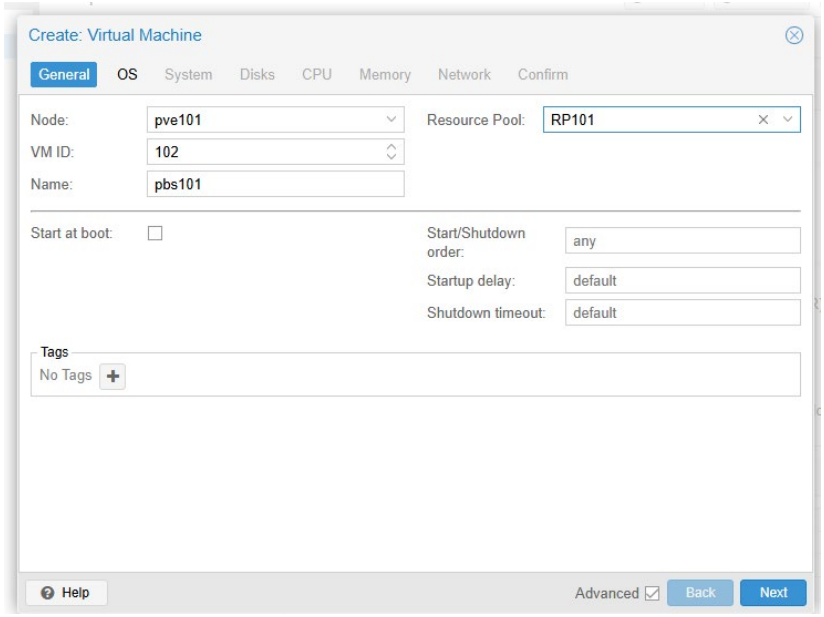
Step 3: VM ID: default

Step 4: Name: pbsXYZ

Step 5: Resource Pool: RPXYZ

Step 6: Next

e.g.:



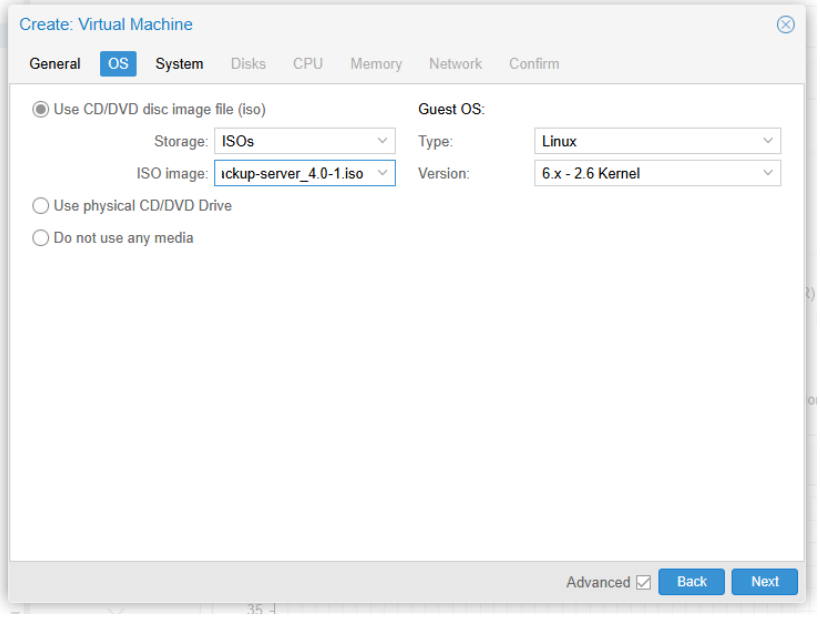
Step 7: Storage: ISOs

Step 8: ISO Image: Proxmox-backup-server...

Step 9: Type: Linux

Step 10: Version: 6.x – 2.6 Kernel

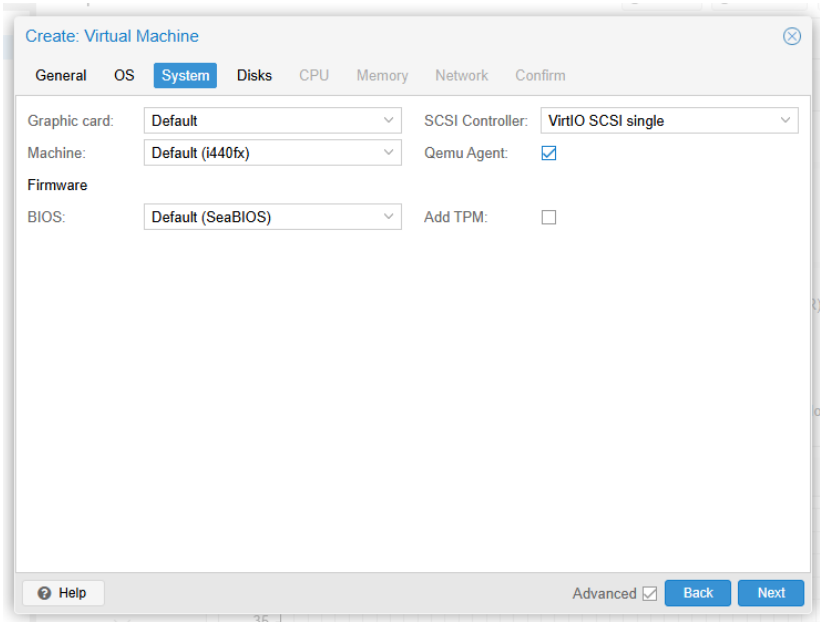
e.g.:



Step 11: All defaults

Step 12: Qemu Agent: checked

Step 13:

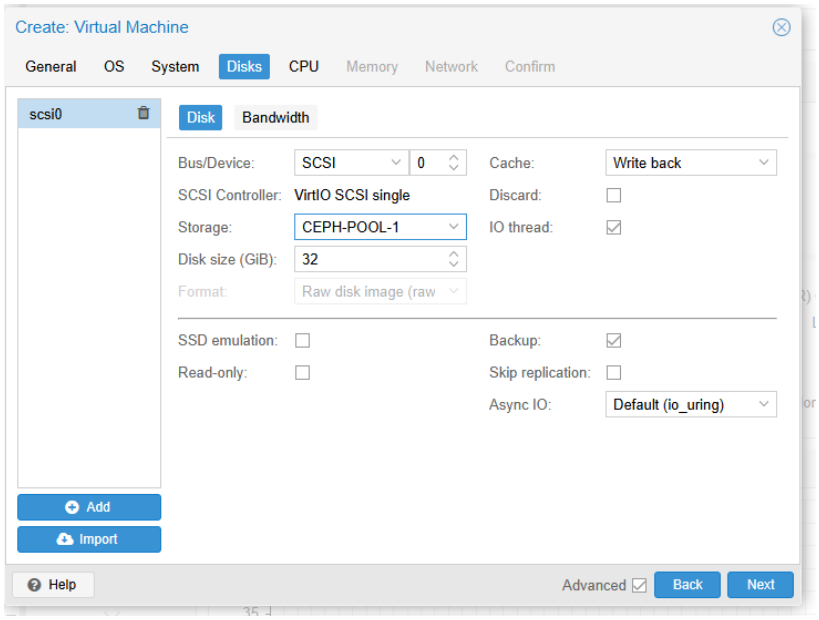


Step 14: Defaults except

Step 15: Storage: Any valid pool

Step 16: Cache: Write back

e.g.:

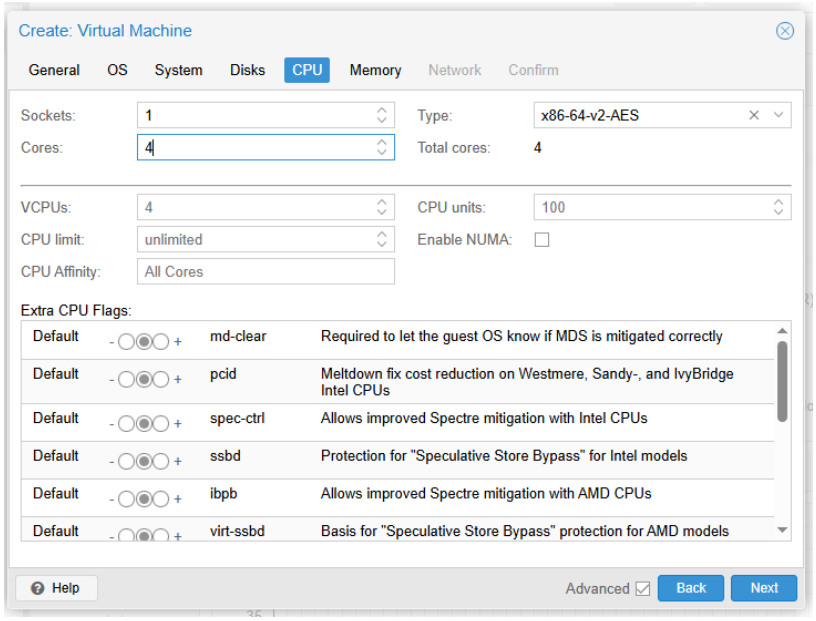


Step 17: Defaults except

Step 18: Sockets: 1

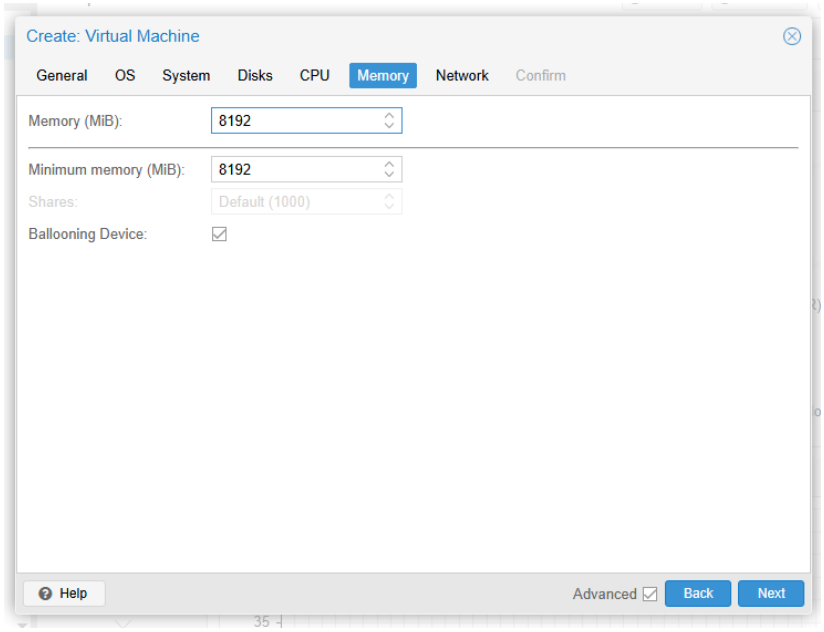
Step 19: Cores: 4

e.g.:



Step 20: Memory: 8192

e.g.:

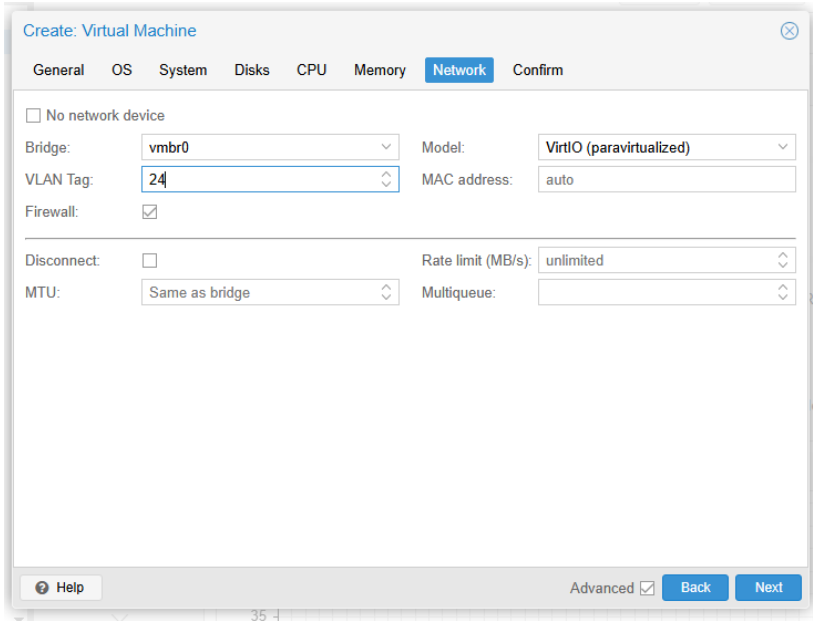


Step 21: Defaults except:

Step 22: Bridge: vmbr0

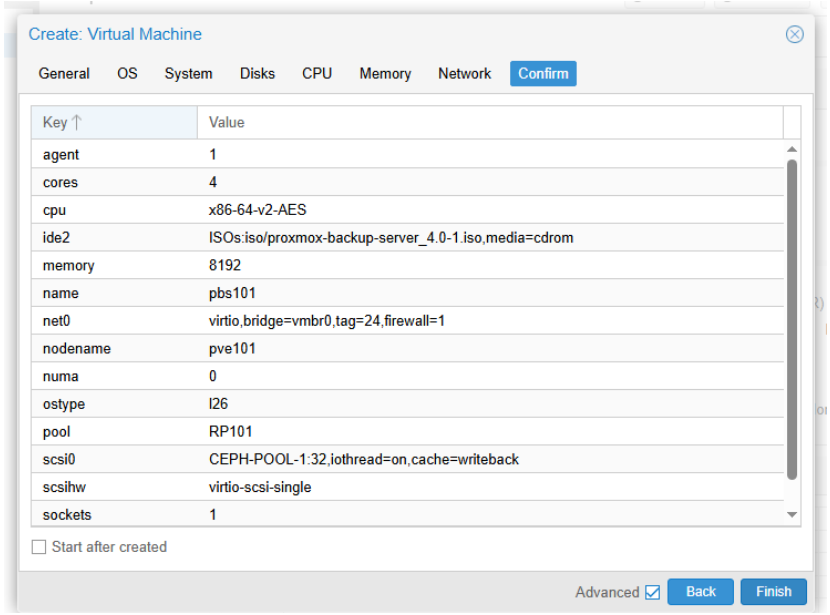
Step 23: VLAN Tag: 24

e.g.:



Step 24: Finish

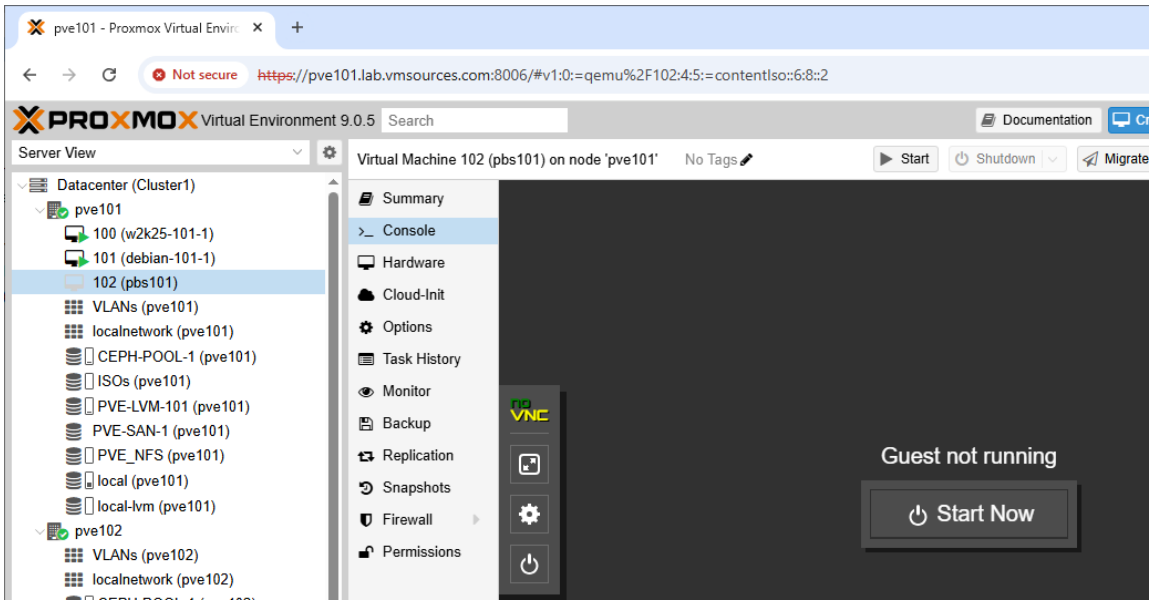
e.g.:



2. Install PBS on VM

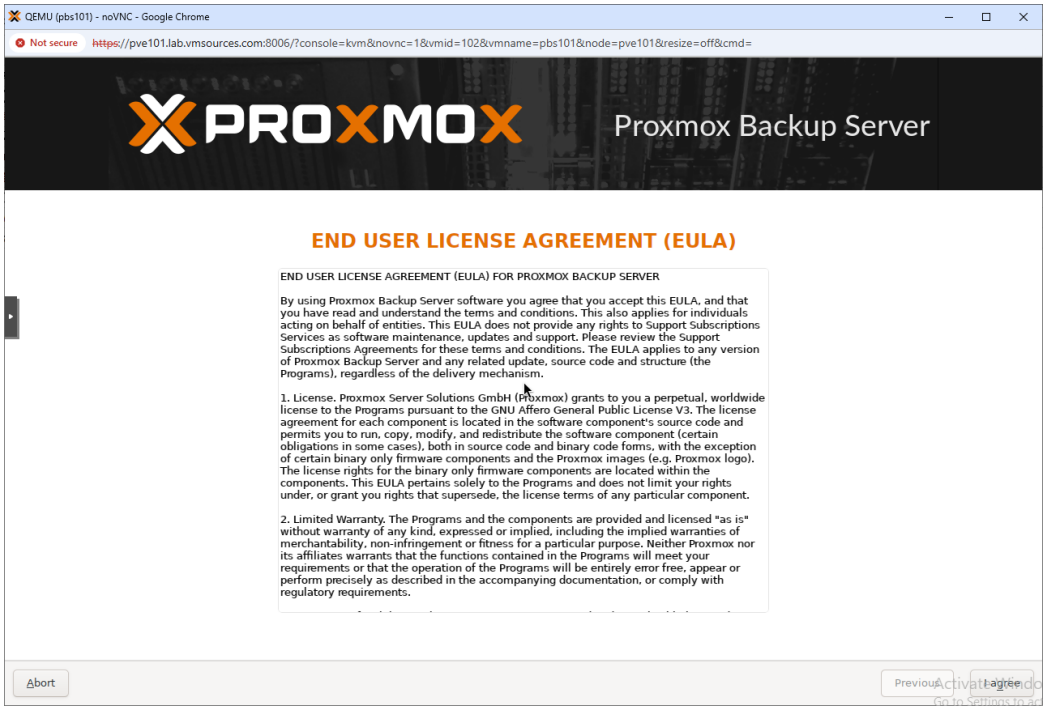
Step 1: pveXYZ > pbsXYZ > Console > Start Now

e.g.:



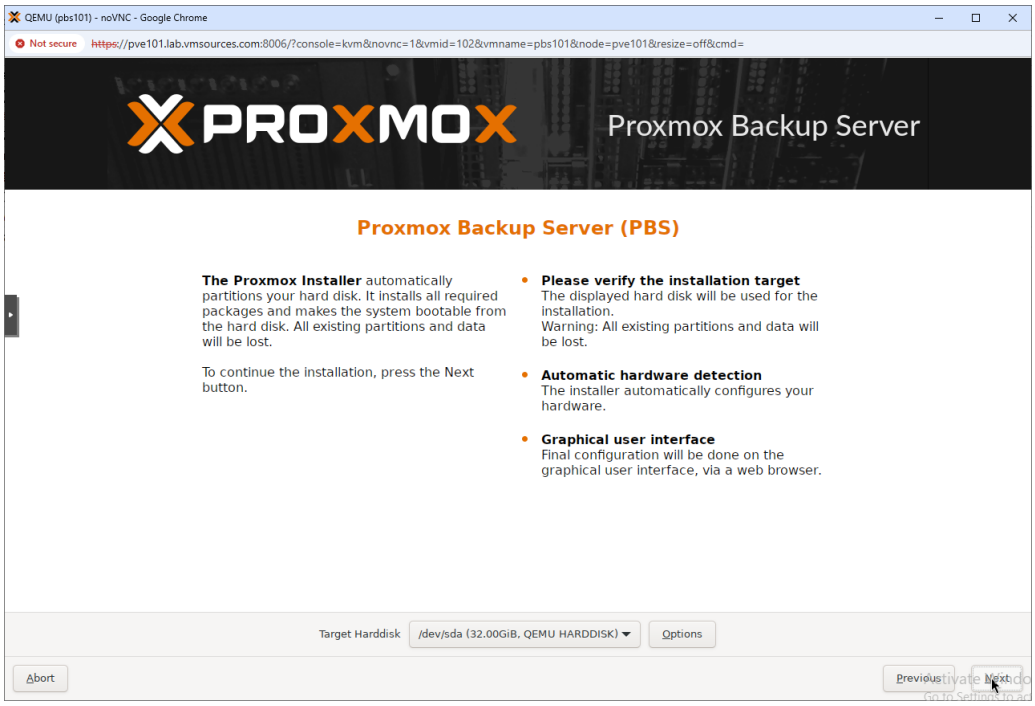

Proxmox Backup Server 4.0 (iso release 1) - https://www.proxmox.com/
PROXMOX
Welcome to Proxmox Backup Server
Install Proxmox Backup Server (Graphical)
Install Proxmox Backup Server (Terminal UI)
Advanced Options

Step 2:

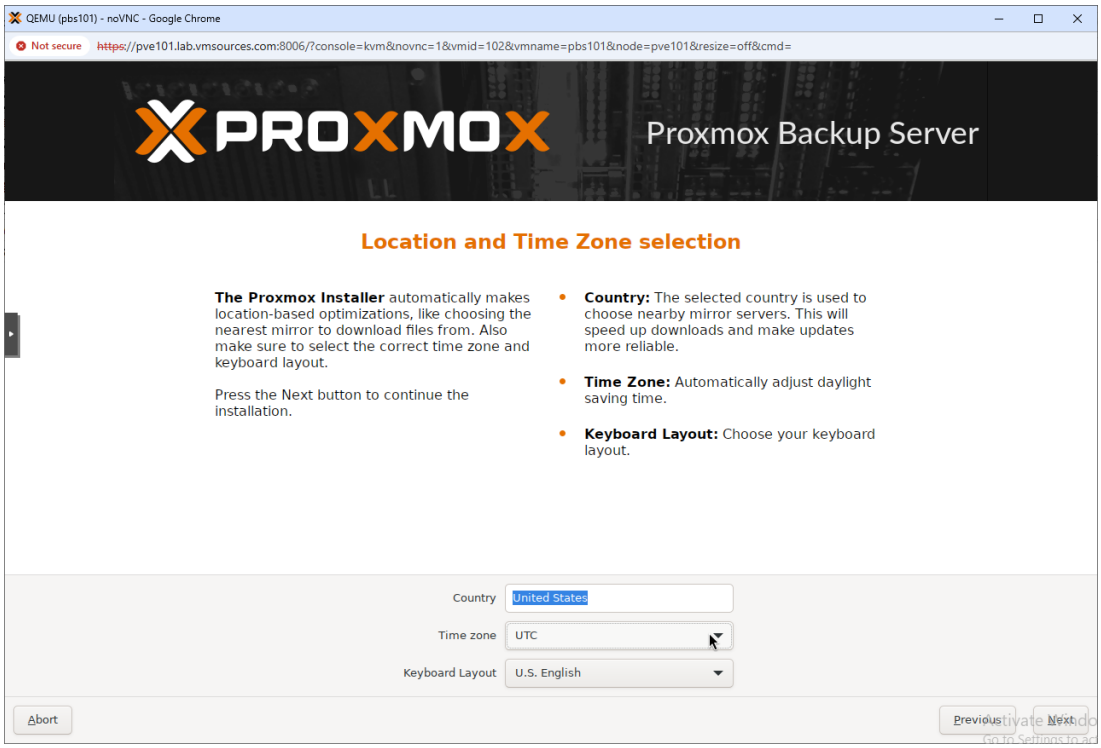

QEMU (pbs101) - noVNC - Google Chrome
Not secure
https://pve101.lab.vmsources.com:8006/?console=kvm&novnc=1&vmid=102&vmname=pbs101&node=pve101&resize=off&cmd=
PROXMOX
Proxmox Backup Server
END USER LICENSE AGREEMENT (EULA)
END USER LICENSE AGREEMENT (EULA) FOR PROXMOX BACKUP SERVER
By using Proxmox Backup Server software you agree that you accept this EULA, and that you have read and understand the terms and conditions. This also applies for individuals acting on behalf of entities. This EULA does not provide any rights to Support Subscriptions Services as software maintenance, updates and support. Please review the Support Subscriptions Agreements for these terms and conditions. The EULA applies to any version of Proxmox Backup Server and any related update, source code and structure (the Programs), regardless of the delivery mechanism.
1. License. Proxmox Server Solutions GmbH (Proxmox) grants to you a perpetual, worldwide license to the Programs pursuant to the GNU Affero General Public License V3. The license agreement for each component is located in the software component's source code and permits you to run, copy, modify, and redistribute the software component (certain obligations in some cases), both in source code and binary code forms, with the exception of certain binary only firmware components and the Proxmox images (e.g. Proxmox logo). The license rights for the binary only firmware components are located within the components. This EULA pertains solely to the Programs and does not limit your rights under, or grant you rights that supersede, the license terms of any particular component.
2. Limited Warranty. The Programs and the components are provided and licensed "as is" without warranty of any kind, expressed or implied, including the implied warranties of merchantability, non-infringement or fitness for a particular purpose. Neither Proxmox nor its affiliates warrants that the functions contained in the Programs will meet your requirements or that the operation of the Programs will be entirely error free, appear or perform precisely as described in the accompanying documentation, or comply with regulatory requirements.
Abort
Previous
I agree

Step 3:

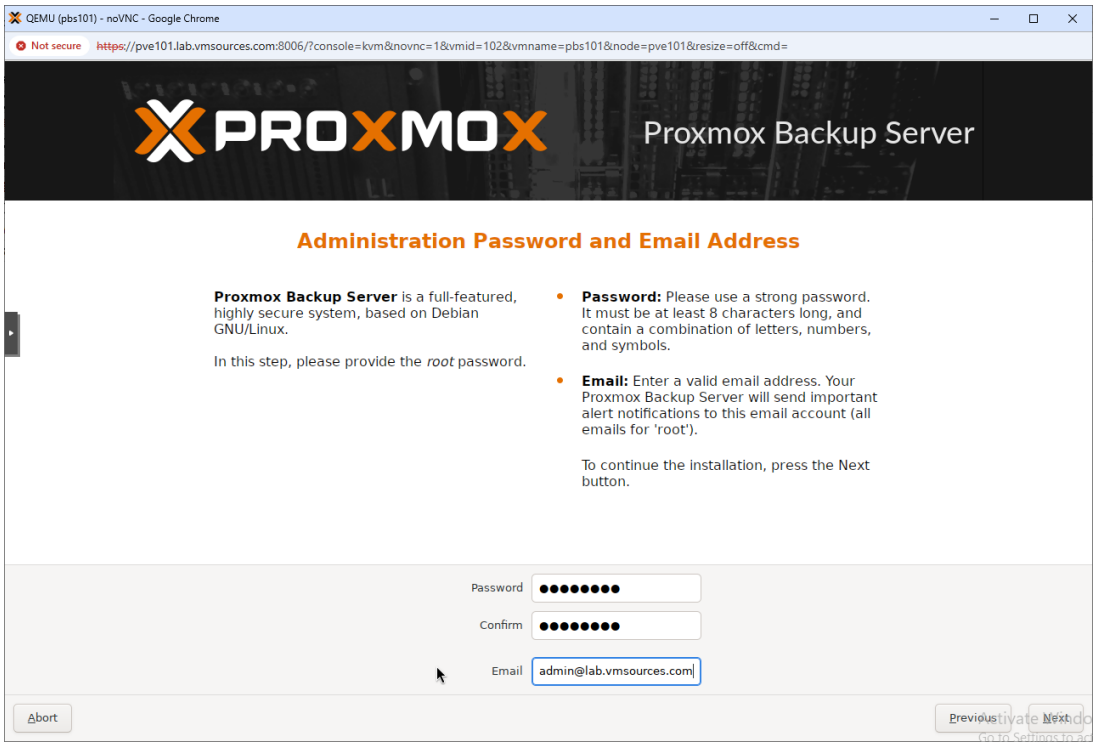
Step 4:



Step 5:

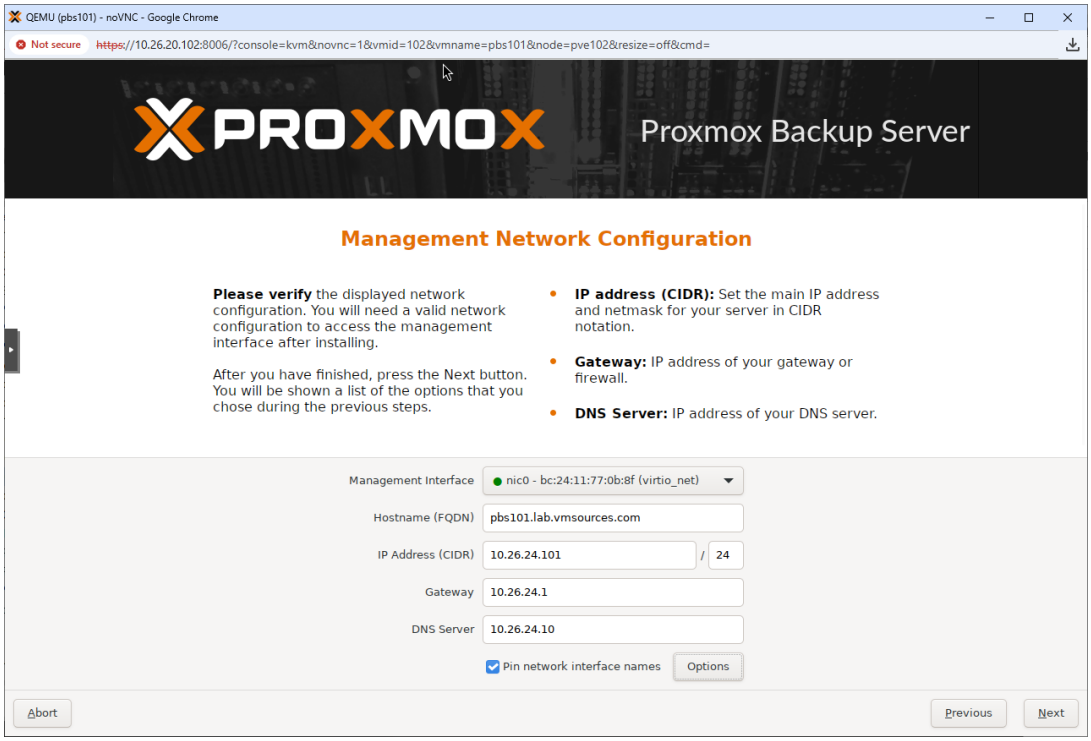


Step 6:


QEMU (pbs101) - noVNC - Google Chrome
Not secure
https://pve101.lab.vmsources.com:8006/?console=kvm&novnc=1&vmid=102&vmname=pbs101&node=pve101&resize=off&cmd=
PROXMOX
Proxmox Backup Server
Administration Password and Email Address
Proxmox Backup Server is a full-featured, highly secure system, based on Debian GNU/Linux.
In this step, please provide the root password.
Password: Please use a strong password. It must be at least 8 characters long, and contain a combination of letters, numbers, and symbols.
Email: Enter a valid email address. Your Proxmox Backup Server will send important alert notifications to this email account (all emails for 'root').
To continue the installation, press the Next button.
Password
Confirm
Email
admin@lab.vmsources.com
Abort
Previous
Next

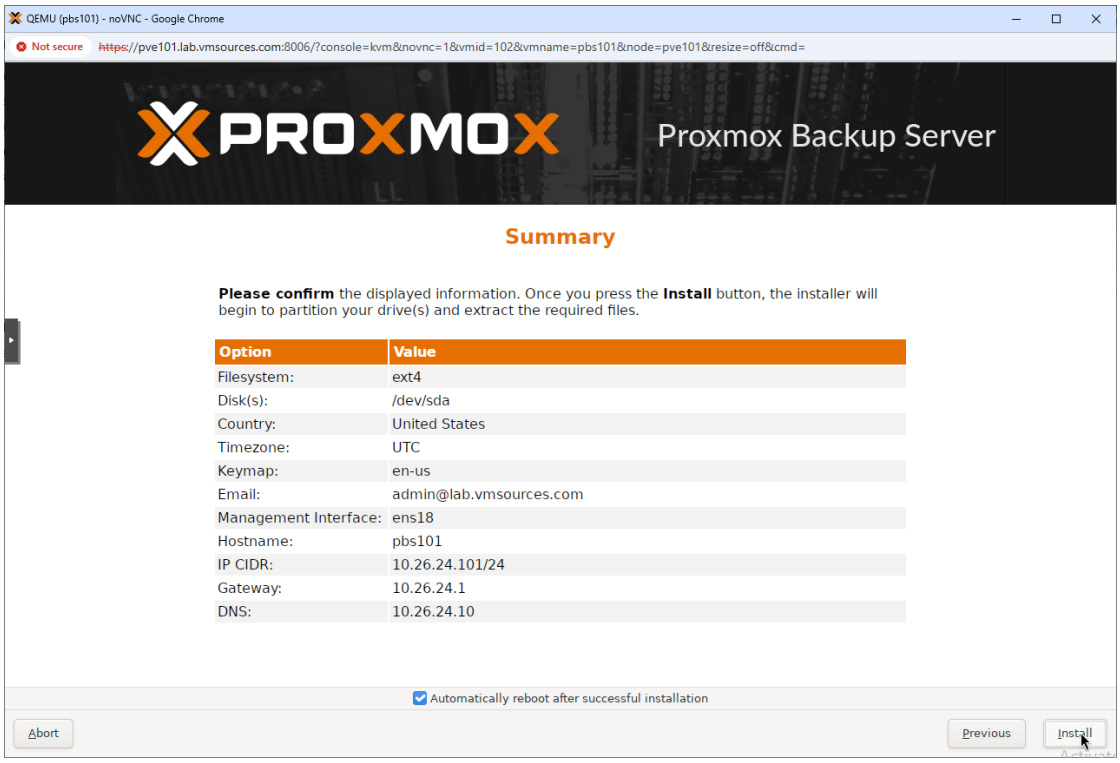
Step 7: Management Interface: nic0 (ens18)
Step 8: Hostname (FQDN): pbsXYZ.lab.vmsources.com
Step 9: IP Address: 10.26.24.XYZ
Step 10: Gateway: 10.26.24.1
Step 11: DNS: 10.26.24.10

e.g.:

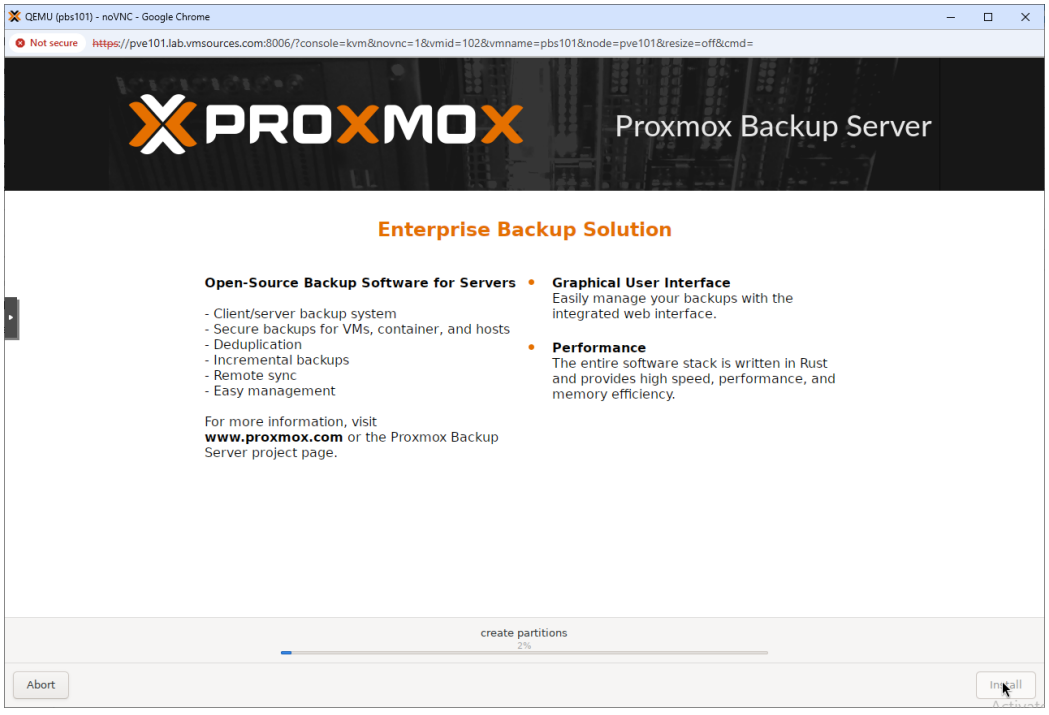


Step 12: Install

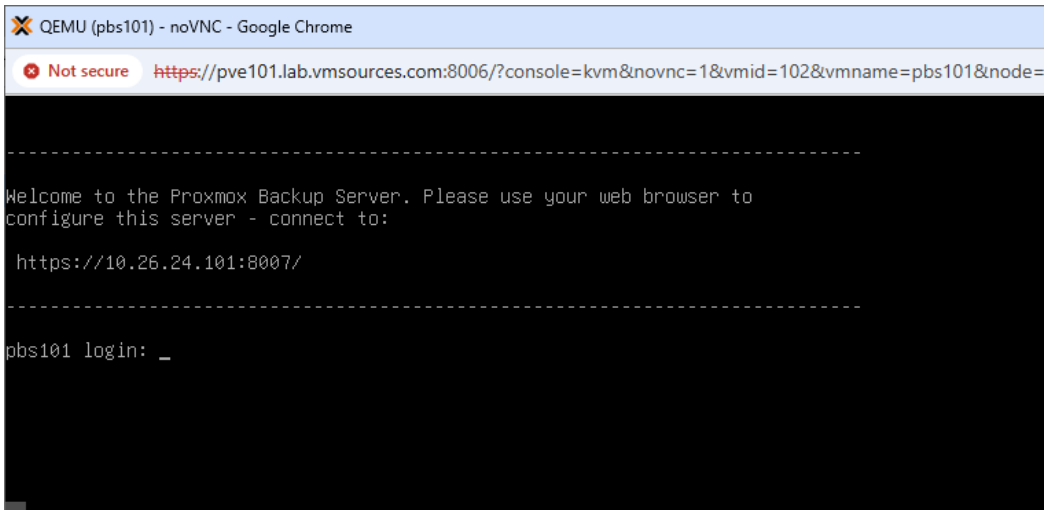
e.g.:


QEMU (pbs101) - noVNC - Google Chrome
Not secure
https://pve101.lab.vmsources.com:8006/?console=kvm&novnc=1&vmid=102&vmname=pbs101&node=pve101&resize=off&cmd=
PROXMOX
Proxmox Backup Server
Summary
Please confirm the displayed information. Once you press the Install button, the installer will begin to partition your drive(s) and extract the required files.
Option
Value
Filesystem:
ext4
Disk(s):
/dev/sda
Country:
United States
Timezone:
UTC
Keymap:
en-us
Email:
admin@lab.vmsources.com
Management Interface:
ens18
Hostname:
pbs101
IP CIDR:
10.26.24.101/24
Gateway:
10.26.24.1
DNS:
10.26.24.10
Automatically reboot after successful installation
Abort
Previous
Install

e.g.:


QEMU (pbs101) - noVNC - Google Chrome
Not secure
https://pve101.lab.vmsources.com:8006/?console=kvm&novnc=1&vmid=102&vmname=pbs101&node=pve101&resize=off&cmd=
PROXMOX
Proxmox Backup Server
Enterprise Backup Solution
Open-Source Backup Software for Servers
- Client/server backup system
- Secure backups for VMs, container, and hosts
- Deduplication
- Incremental backups
- Remote sync
- Easy management
For more information, visit www.proxmox.com or the Proxmox Backup Server project page.
Graphical User Interface
Easily manage your backups with the integrated web interface.
Performance
The entire software stack is written in Rust and provides high speed, performance, and memory efficiency.
create partitions
2%
Abort
Install

e.g.:

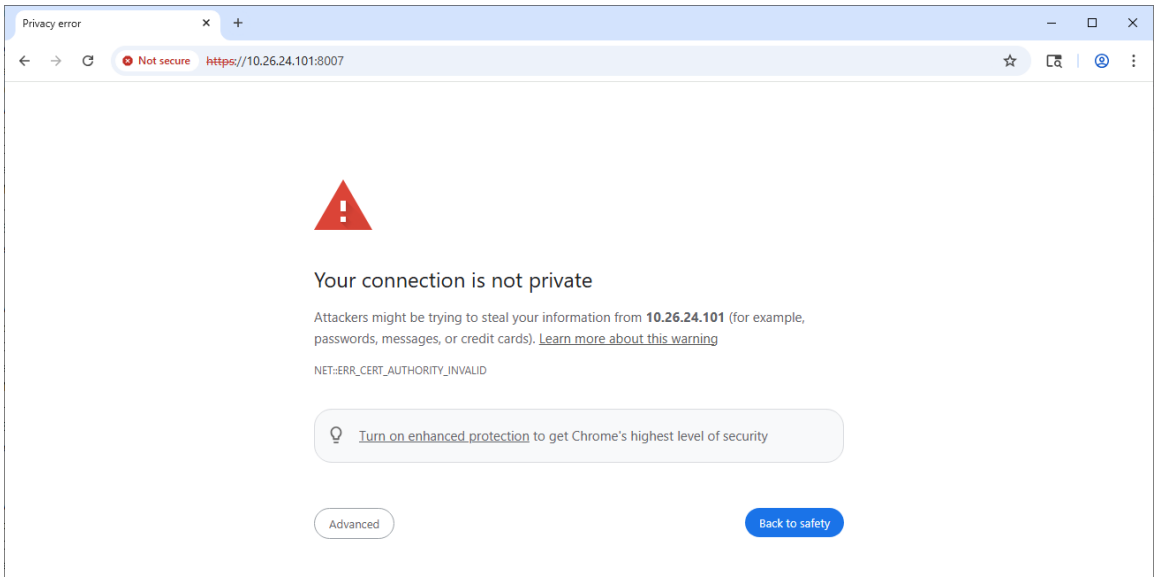

QEMU (pbs101) - noVNC - Google Chrome
Not secure
https://pve101.lab.vmsources.com:8006/?console=kvm&novnc=1&vmid=102&vmname=pbs101&node=
Welcome to the Proxmox Backup Server. Please use your web browser to
configure this server - connect to:
https://10.26.24.101:8007/
pbs101 login:

SBS LAB (GUI) – Login to PBS

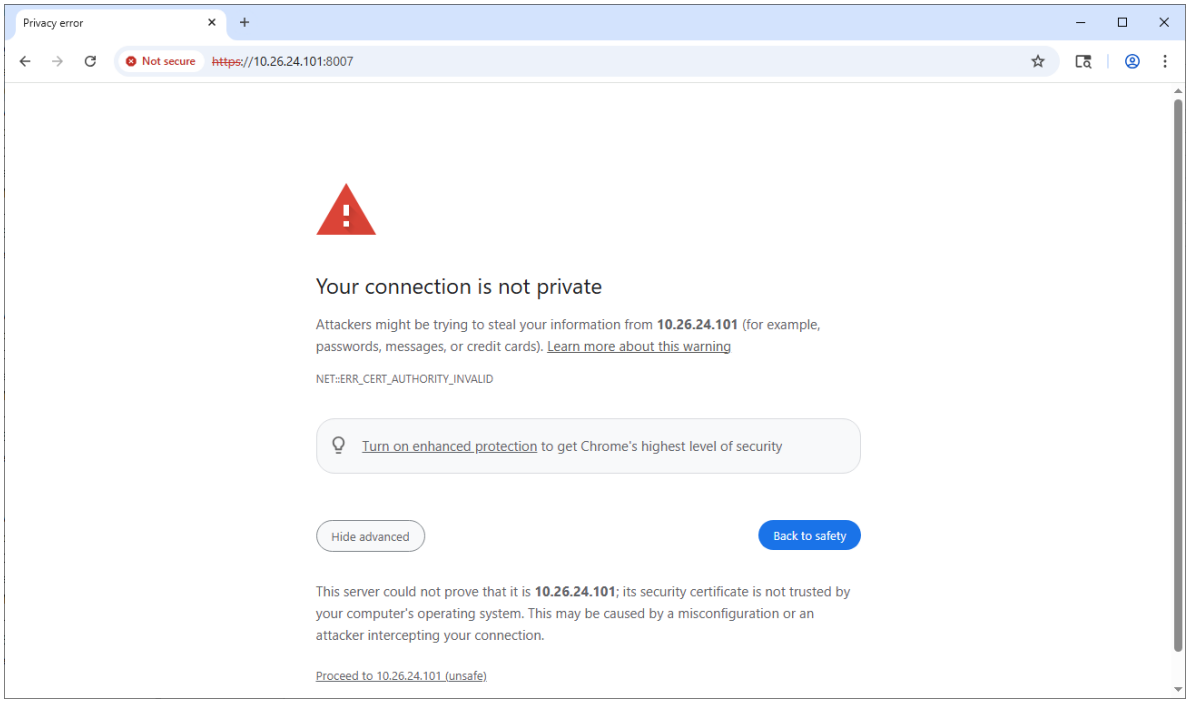
1. Login to PBS

   Step 1: Open a browser to: https://10.26.24.XYZ:8007

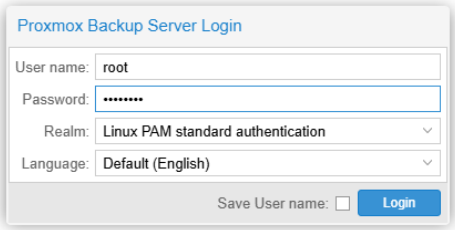
e.g.:



e.g.:



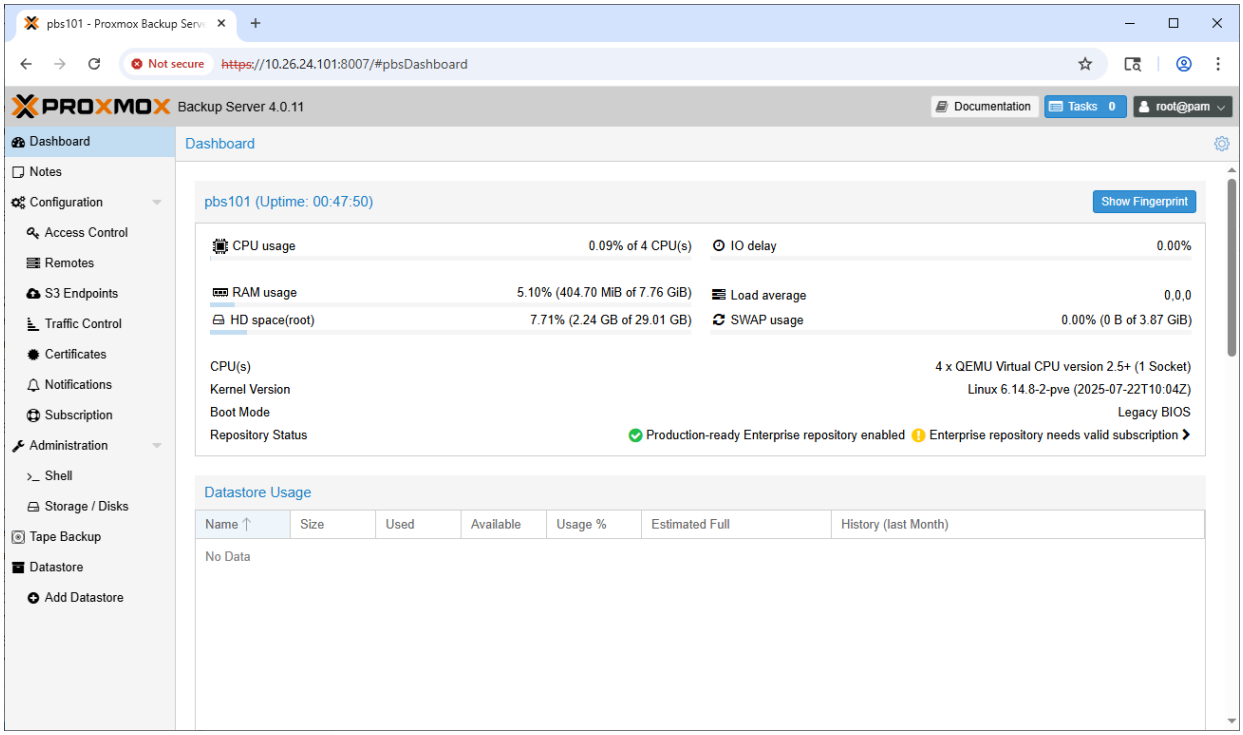
2. Log in



e.g.:

3. You are logged in

e.g.:

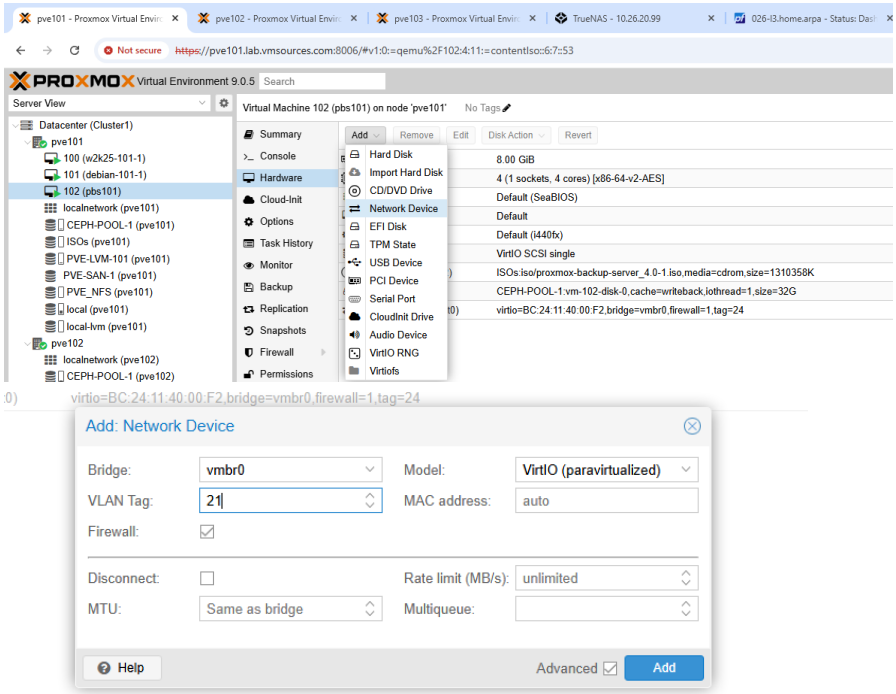


SBS LAB (GUI) – Network Settings for PBS

1. Add a network interface for iSCSI to PBS

   Step 1: pveXYZ > pbsXYZ > Hardware > Add > Network Device

   e.g.:

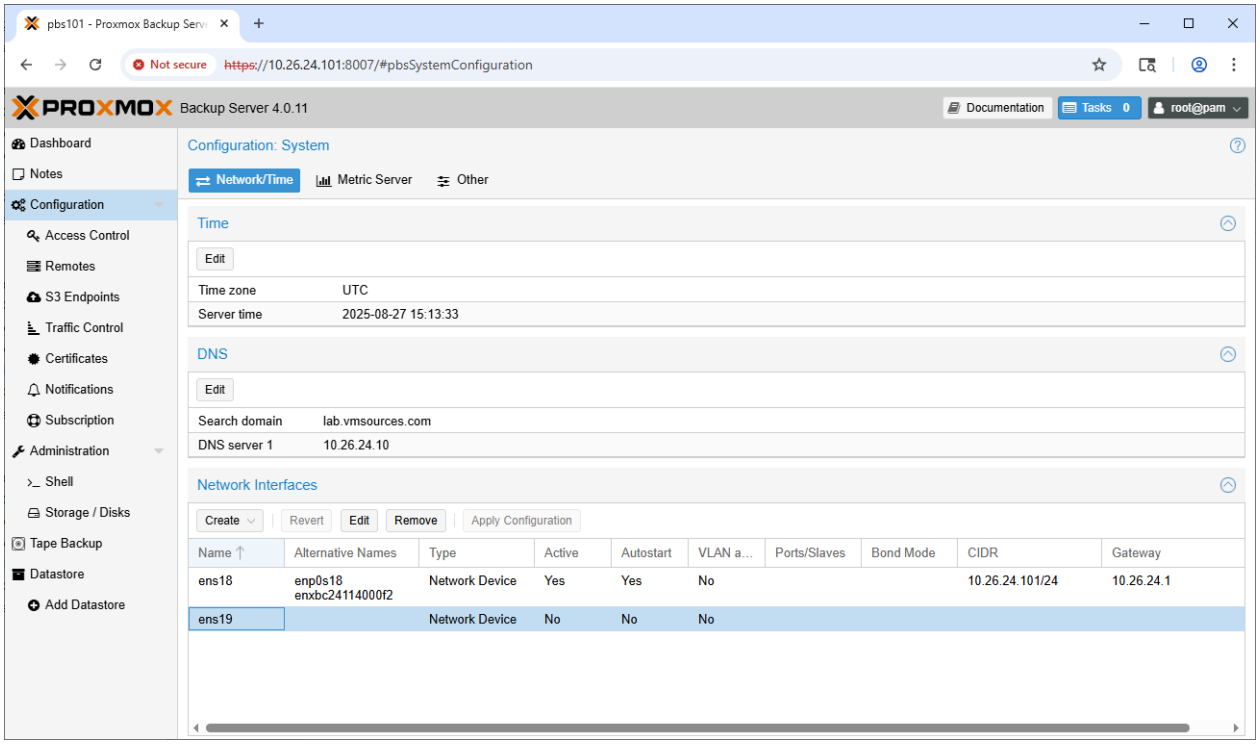


   e.g.:

2. Edit network settings for iSCSI

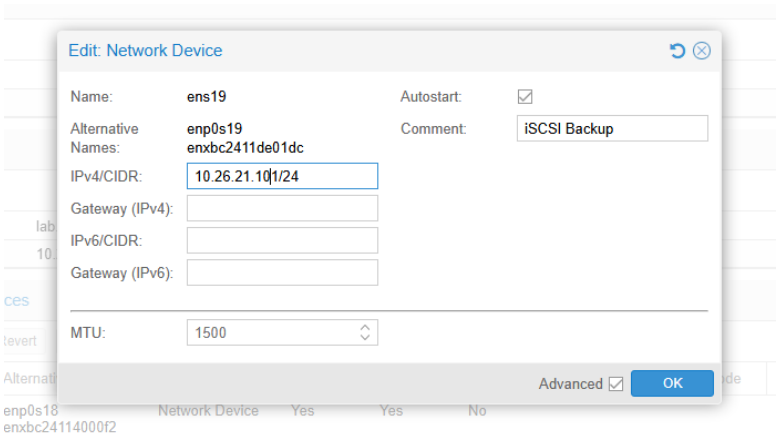
   Step 1: pbsXYZ > Configuration > Network Interfaces > ens19 > Edit

   e.g.:



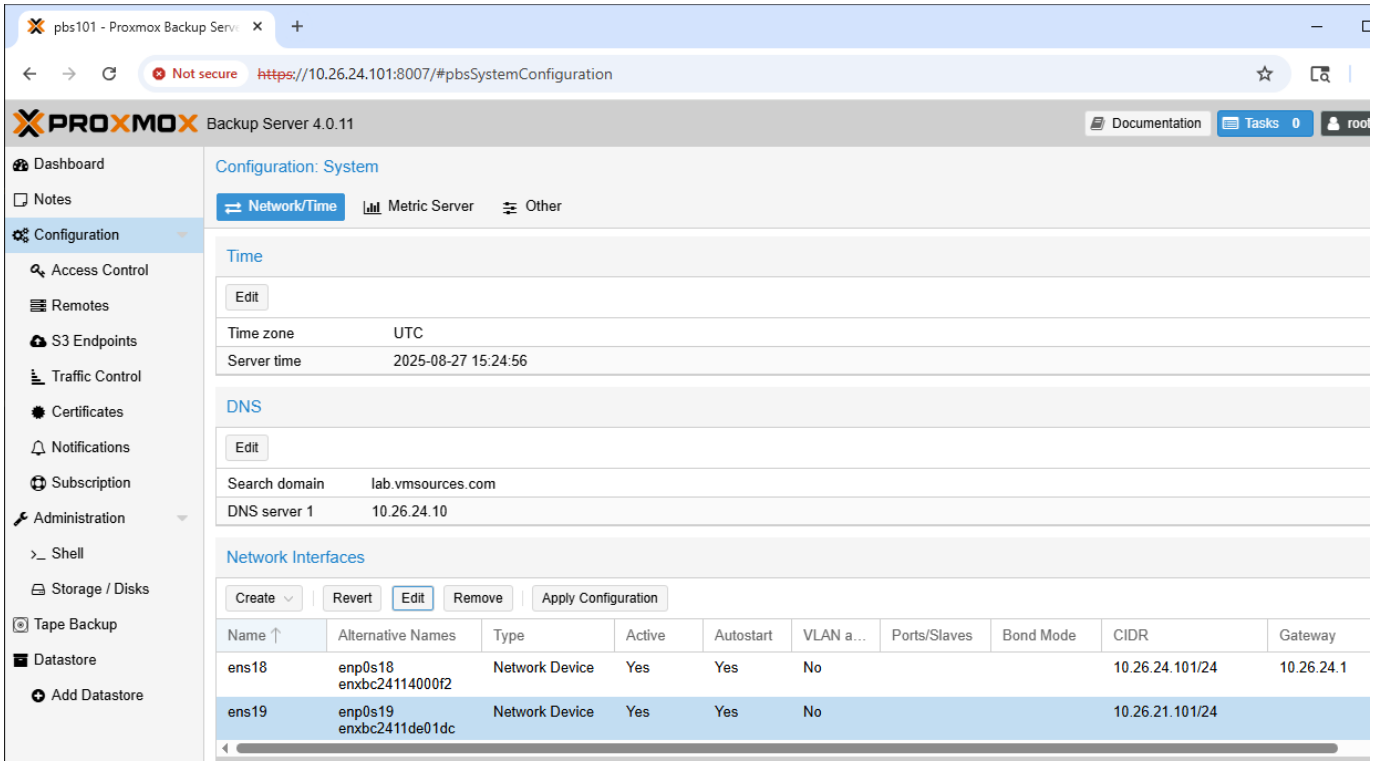
Step 2: Autostart: checked

e.g.: IPv4/CIDR:10.26.21.XYZ

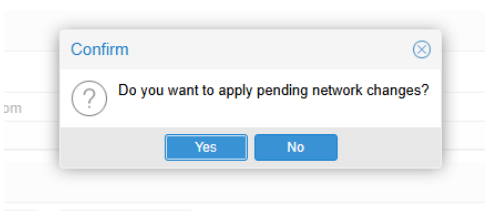


e.g.:

Step 3: Apply Configuration



e.g.:



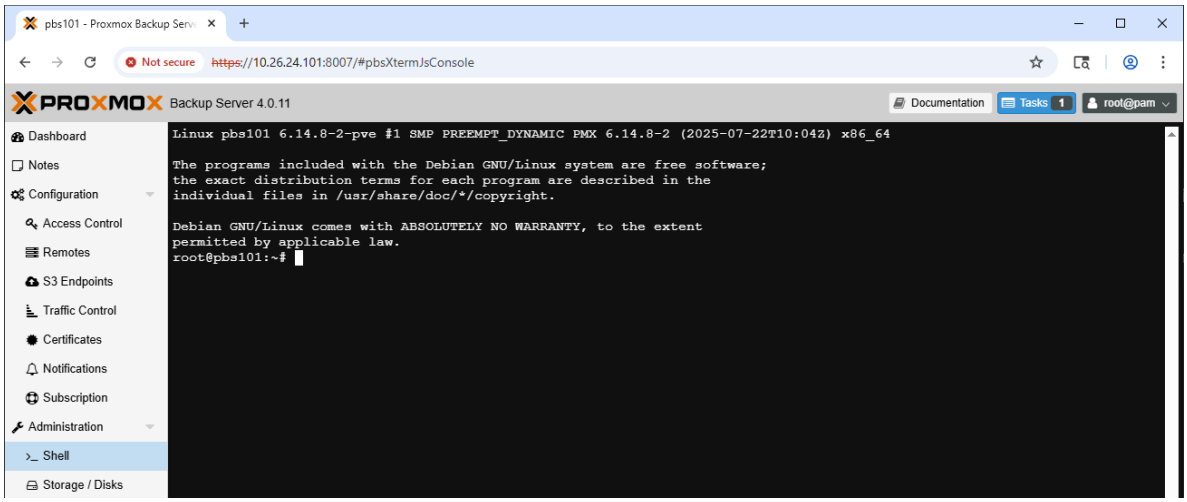
e.g.:

SBS LAB (CLI) – iSCSI for PBS

1. Configure iSCSI Storage

Step 1: Open a shell

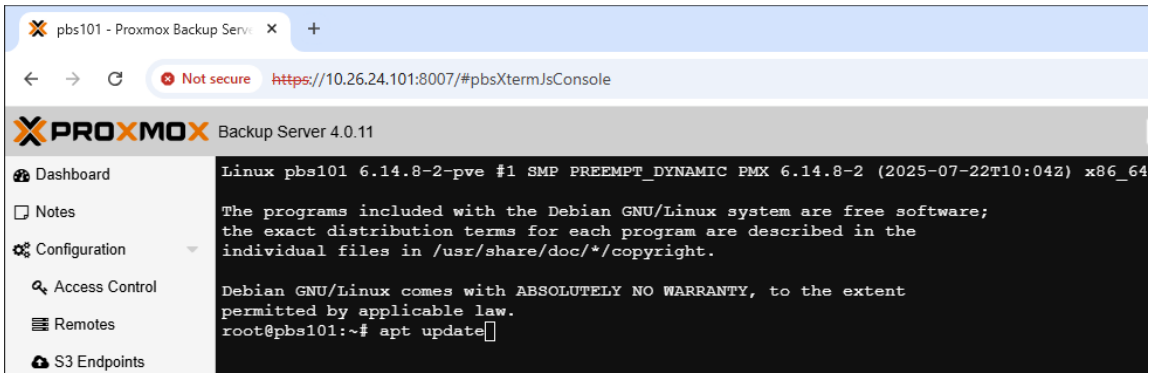
e.g.:



Step 2: Update metadata for available updates

Command #1 run: apt update

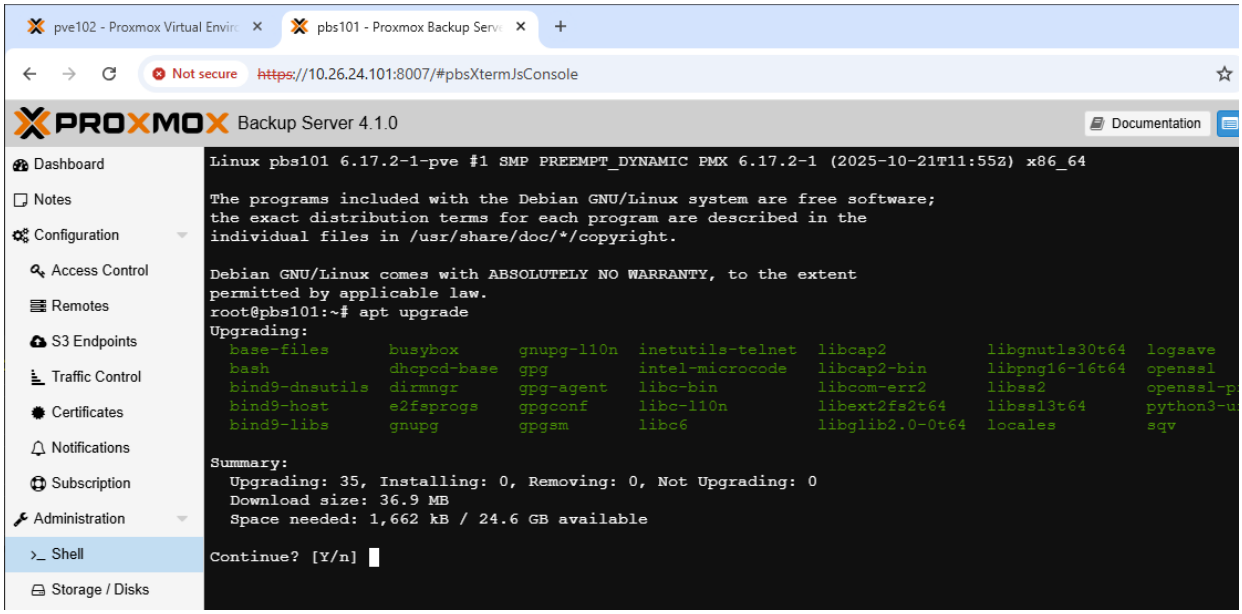
e.g.:



Step 3: Update running system based on metadata

Command #1 run: apt upgrade

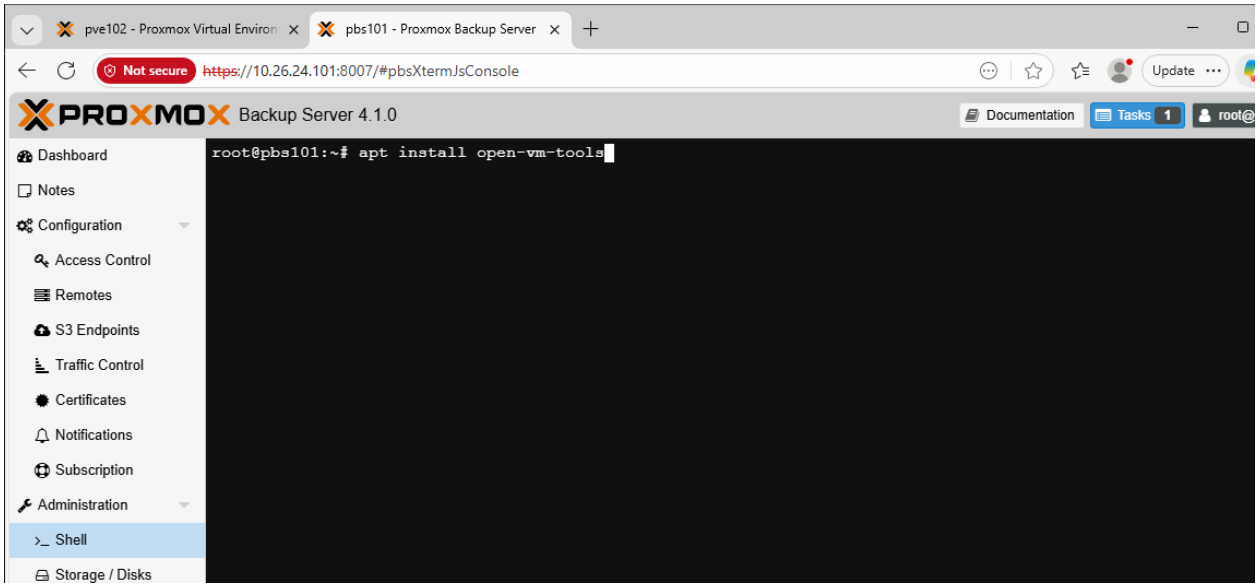
e.g.:



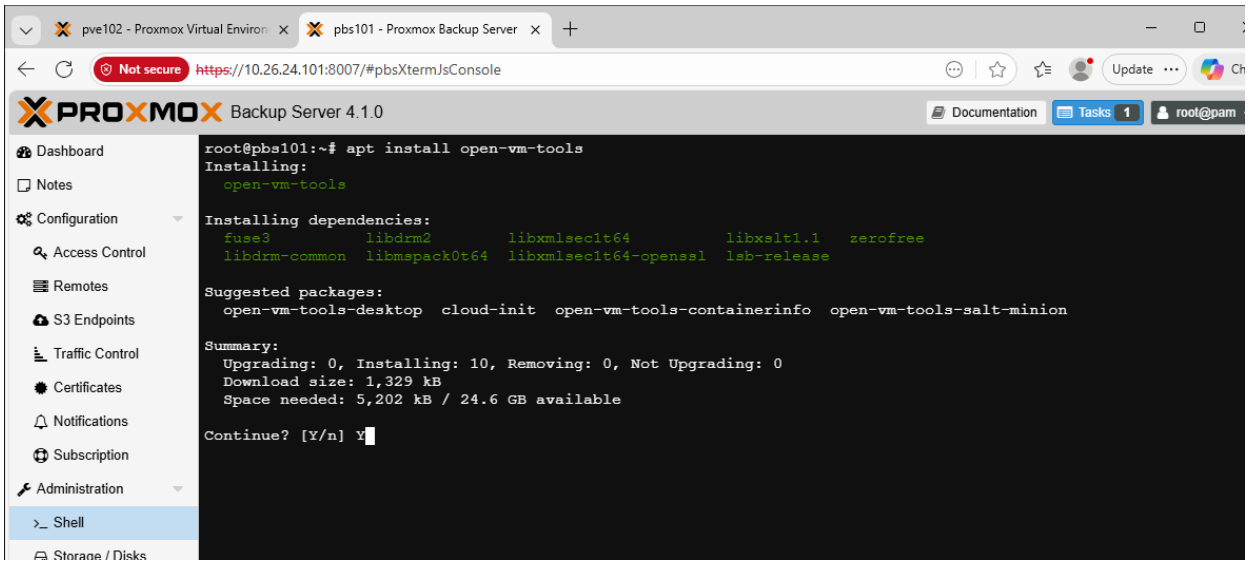
Step 4: Install open-vm-tools

Command #1 run: apt install open-vm-tools

e.g.:



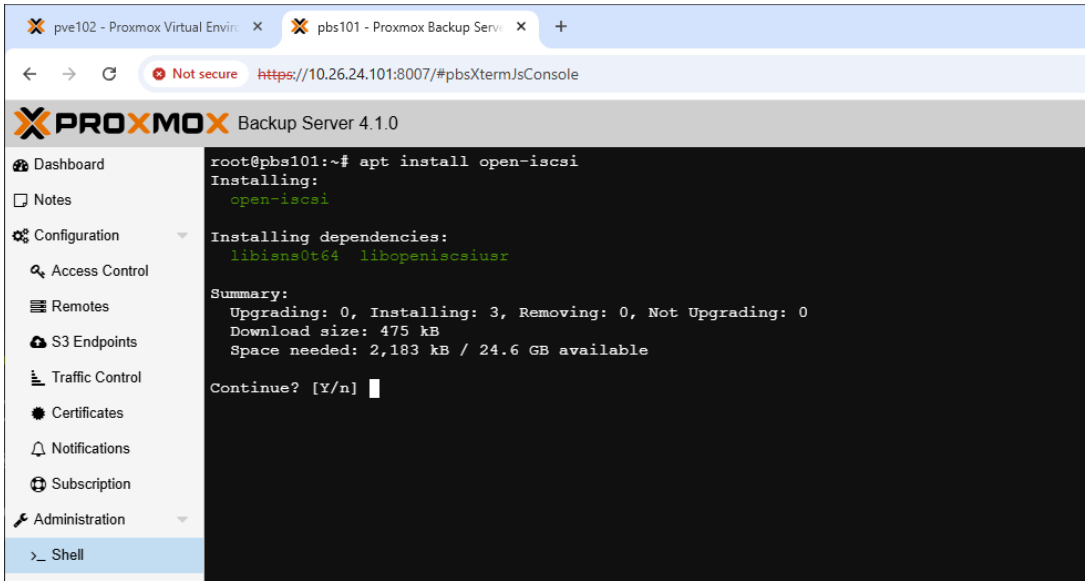
e.g.:



Step 5: Install open-iscsi

Command #1 run: apt install open-iscsi

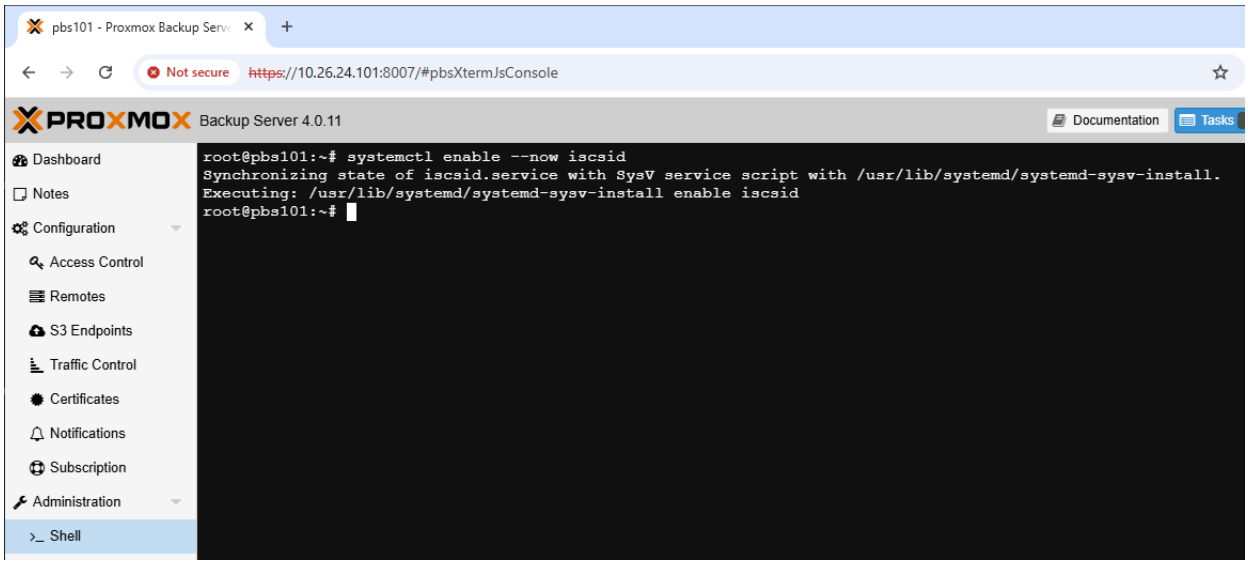
e.g.:



Step 6: Enable the iSCSI daemon

Command #1 run: systemctl enable --now iscsid

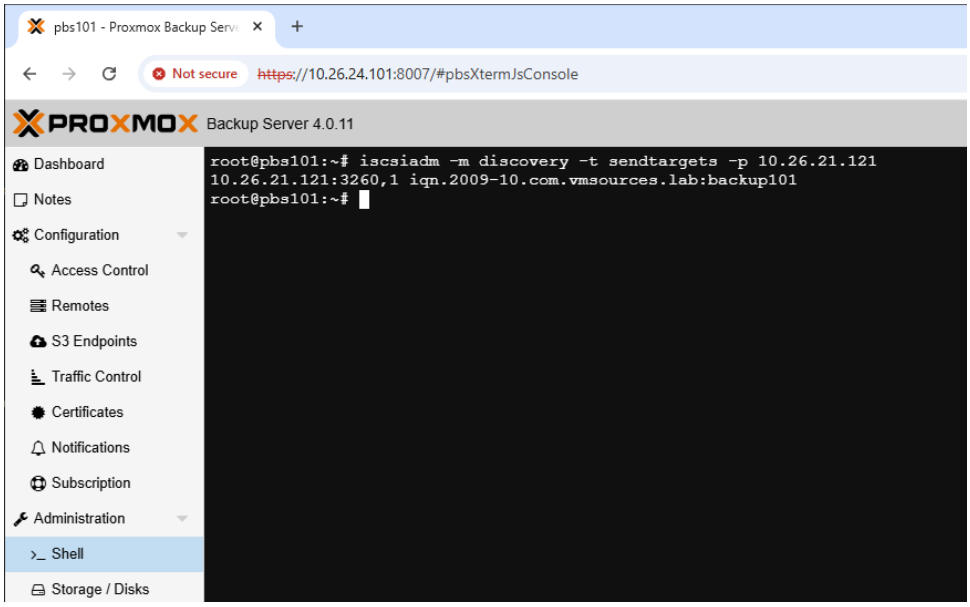
e.g.:



Step 7: Discover your backup LUN

Command #1 run: iscsiadm -m discovery -t sendtargets -p 10.26.21.ABC

e.g.:



Step 8: Now we'll set iSCSI up for automatic login and restart the service

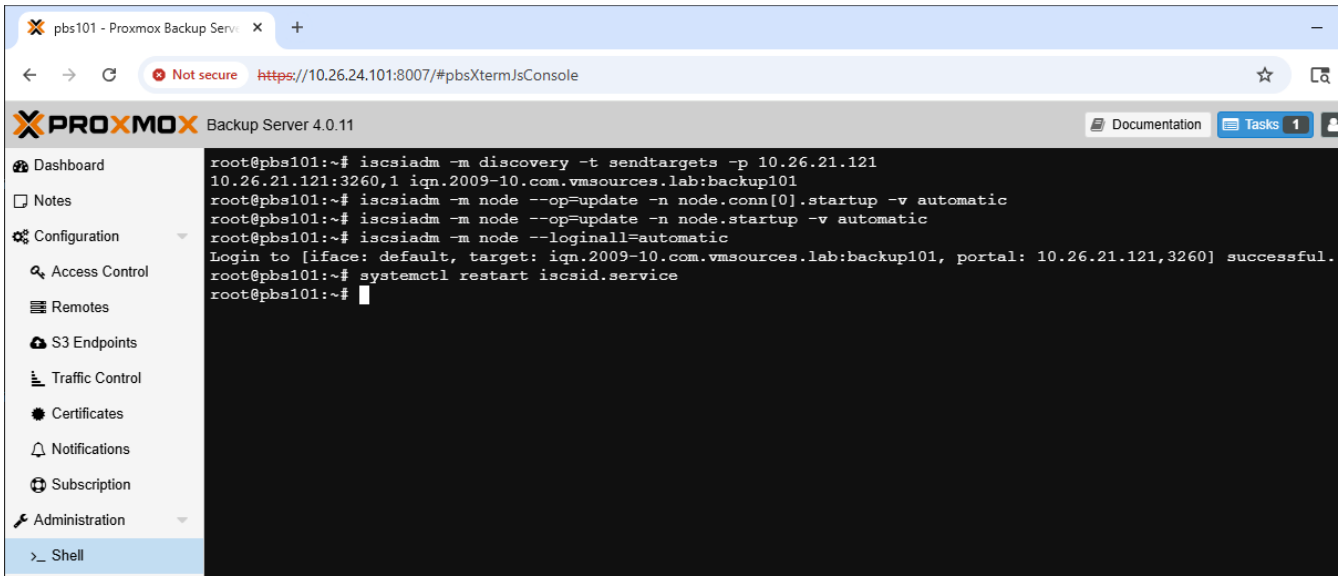
Command #1 run: iscsiadm -m node --op=update -n node.conn[0].startup -v automatic

Command #2 run: iscsiadm -m node --op=update -n node.startup -v automatic

Command #3 run: iscsiadm -m node --loginall=automatic

Command #4 run: systemctl restart iscsid.service

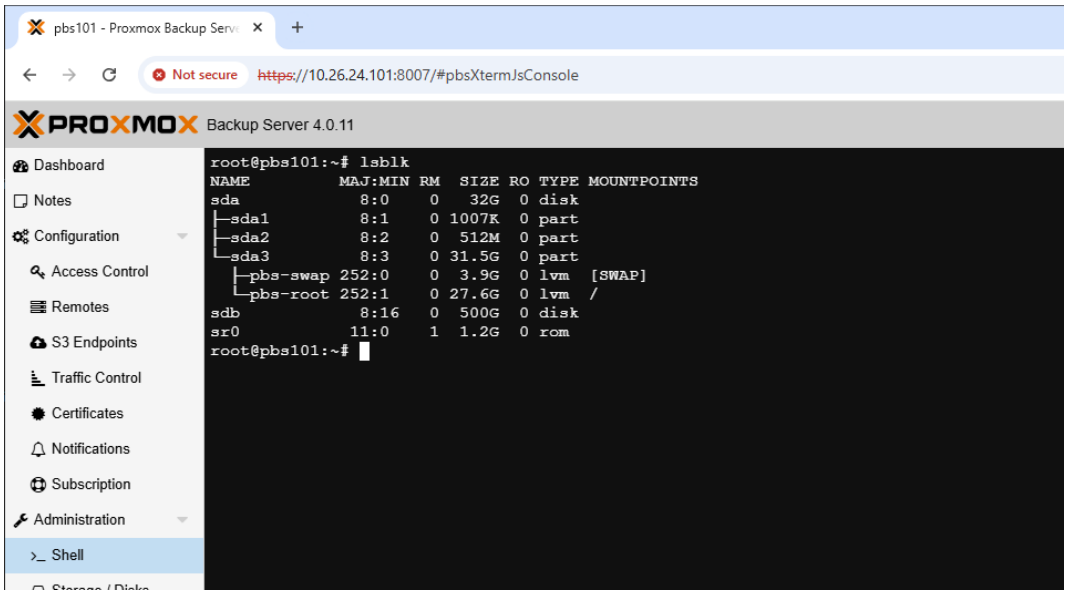
e.g.:



Step 9: List the block devices available

Command #1 run: lsblk

e.g.:

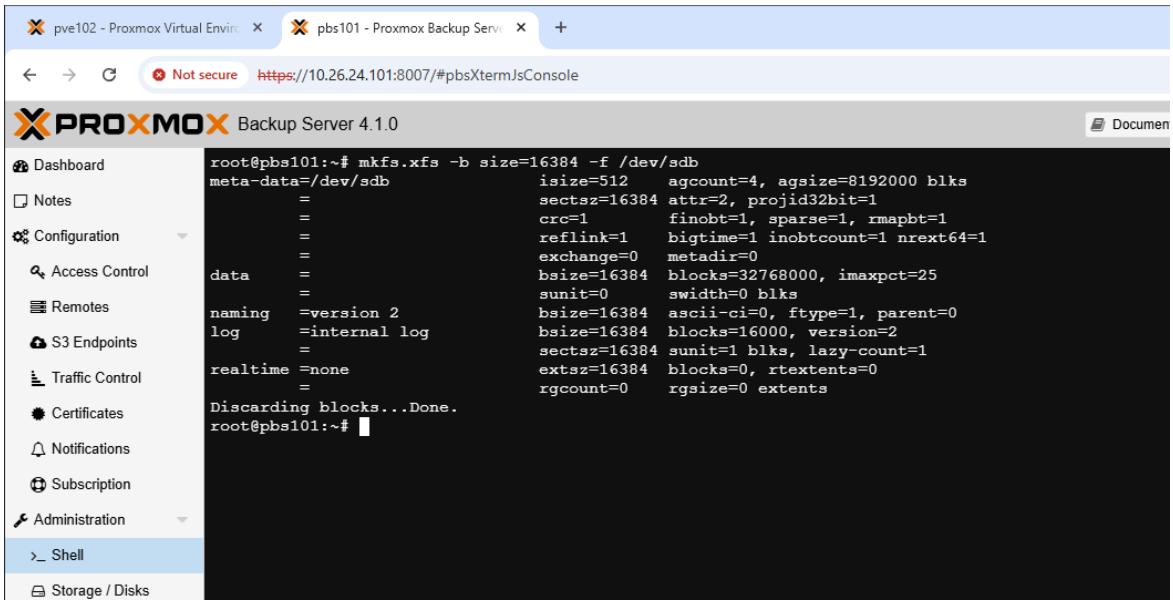


Step 10: Format the iSCSI disk to XFS filesystem:

Command #1 run: mkfs.xfs -b size=16384 -f /dev/sdX

NOTE : Where X is the iSCSI LUN you saw in previous step

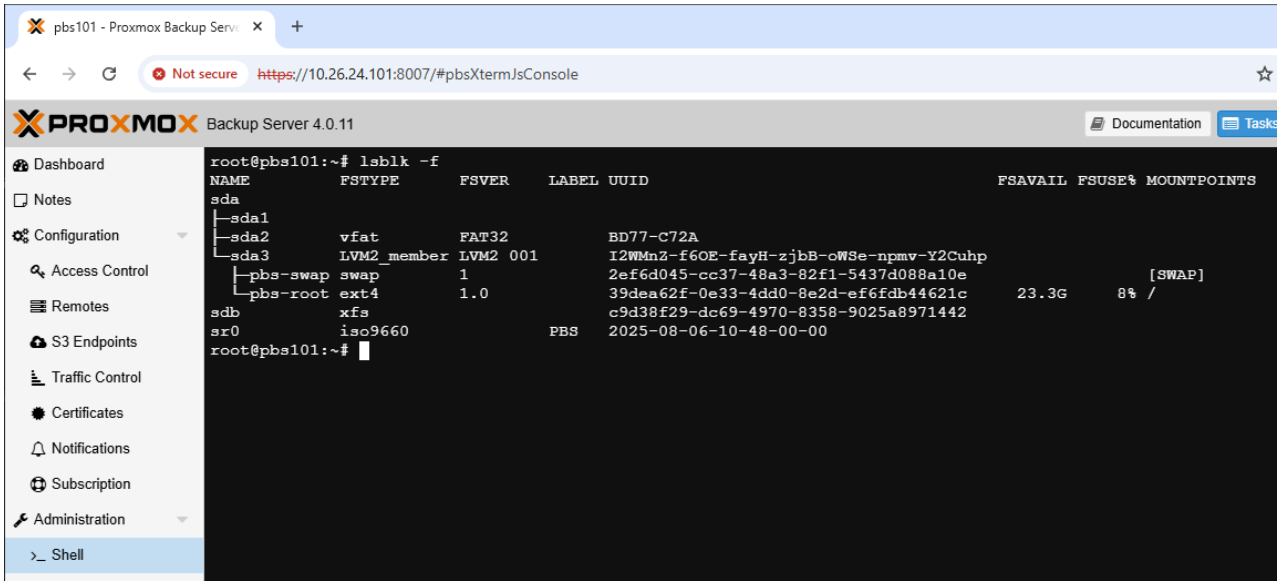
e.g.:



Step 11: Identify the UUID of the iSCSI LUN

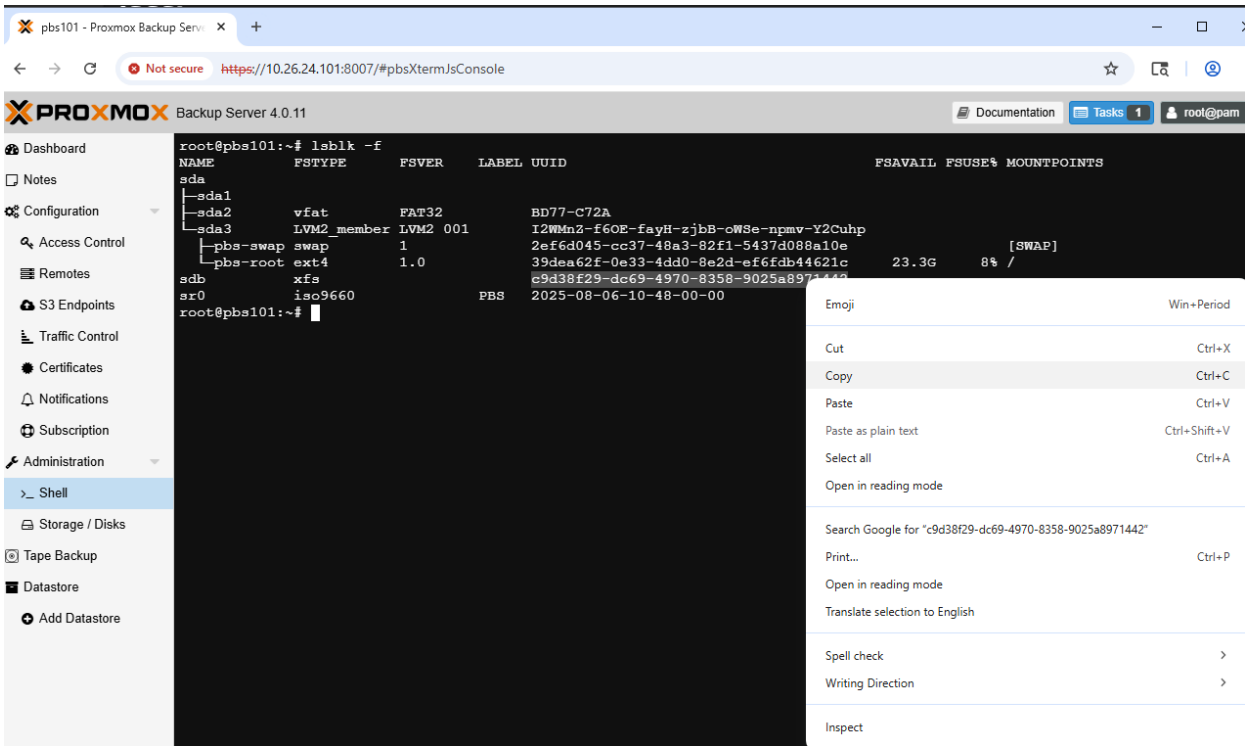
Command #1 run: lsblk -f

e.g.:



Step 12: Copy the UUID of the /dev/sdb volume

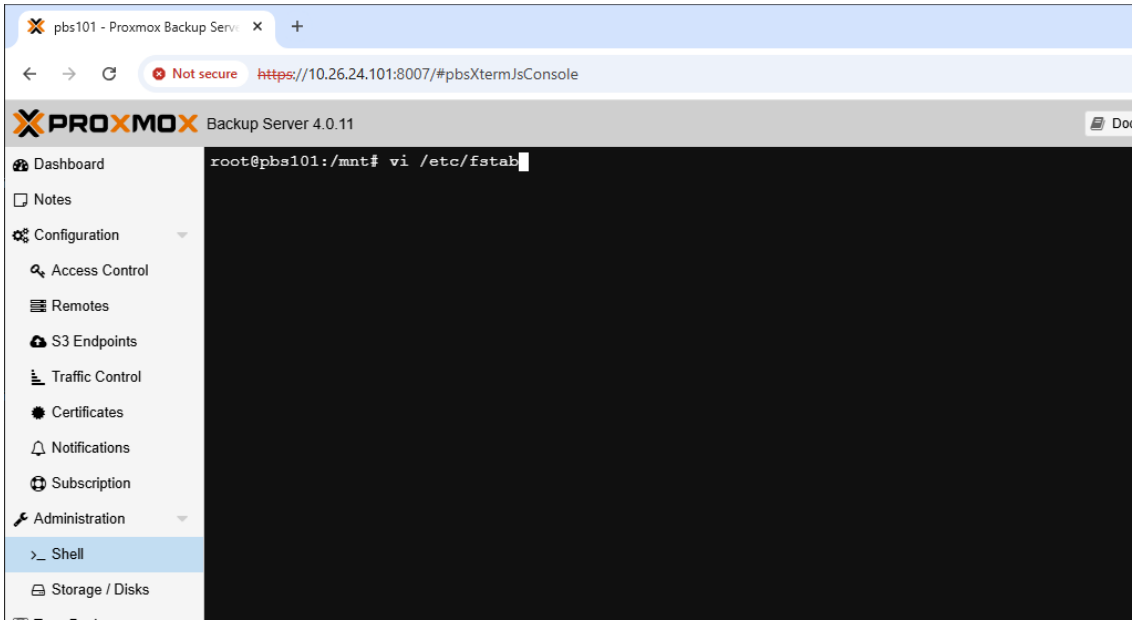
e.g.:



Step 13: Edit /etc/fstab

Command #1 run: vi /etc/fstab

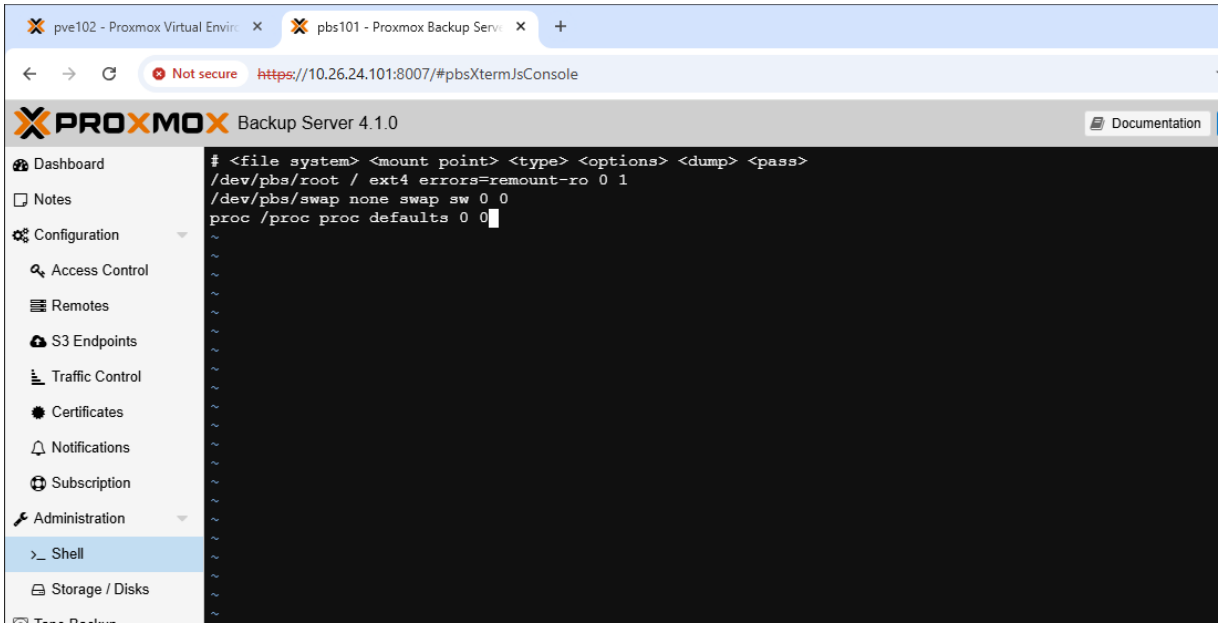
e.g.:



NOTE : /etc/fstab is the file that mounts available block storage at boot

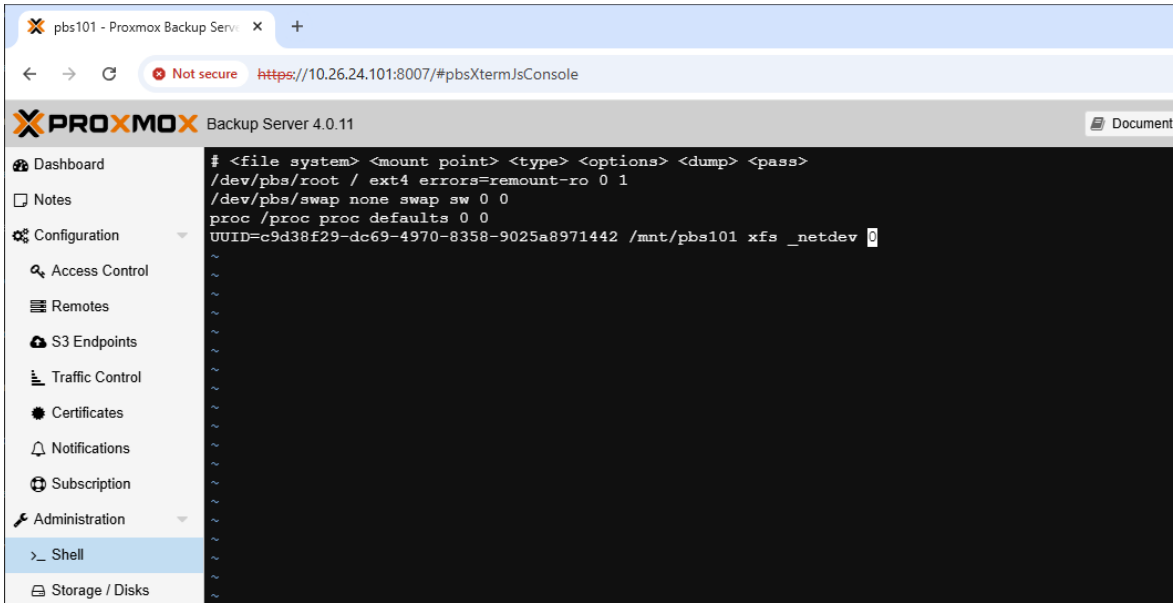
Step 14: Move to the end of the last line and press [a] to enter edit/add mode

e.g.:

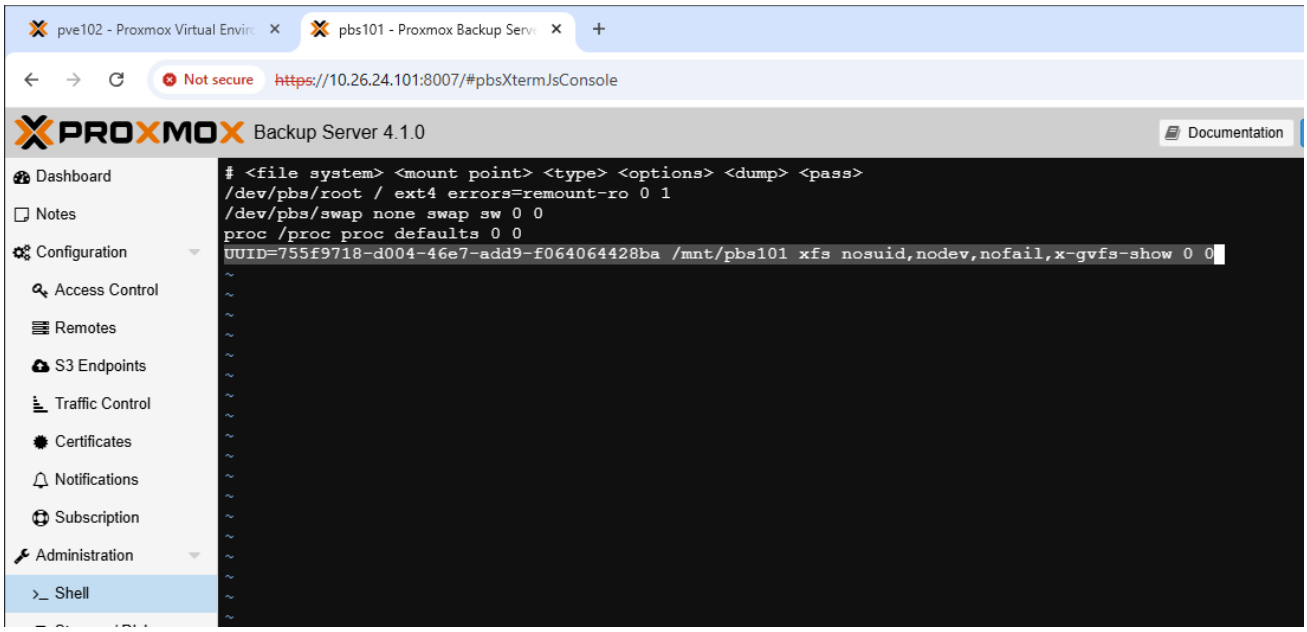


Step 15: Press [Enter] to move to the nest line and paste your UUID like this

e.g.:



e.g.:

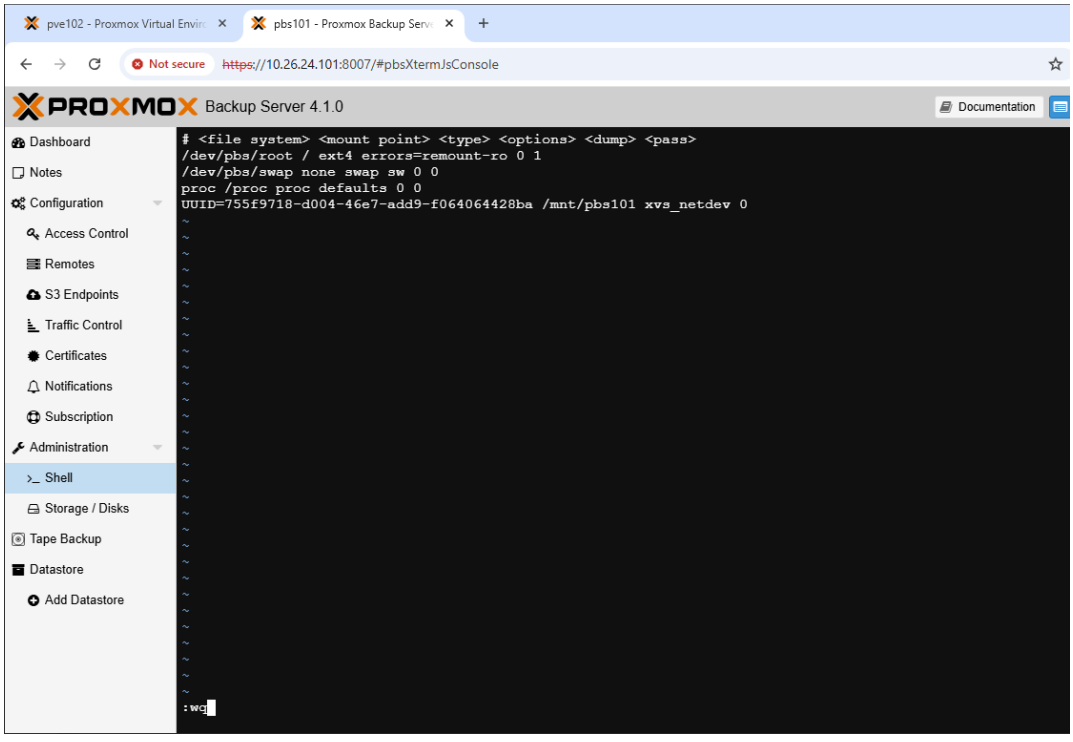


NOTE : Substitute the UUID you copied for the example here and change XYZ to your number:

UUID=755f9718-d004-46e7-add9-f064064428ba /mnt/pbsXYZ xfs nosuid,nodev,nofail,x-gvfs-show 0 0

Step 16: Save the file by pressing [ESC] to exit edit mode and entering ':wq'

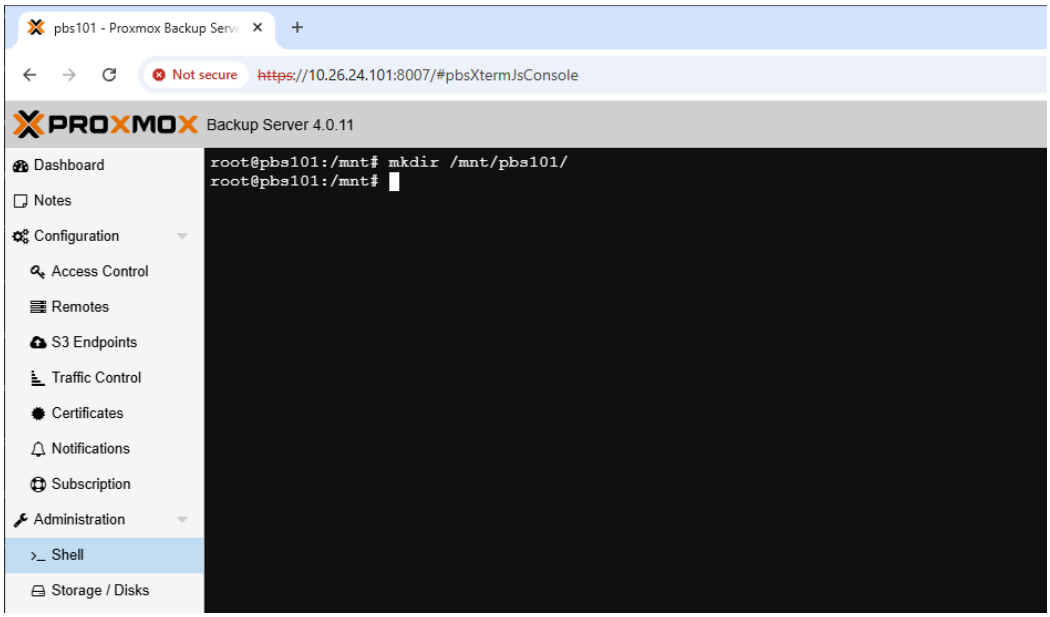
e.g.:



Step 17: Now make a new directory for the mount

Command #1 run: mkdir /mnt/pbsXYZ

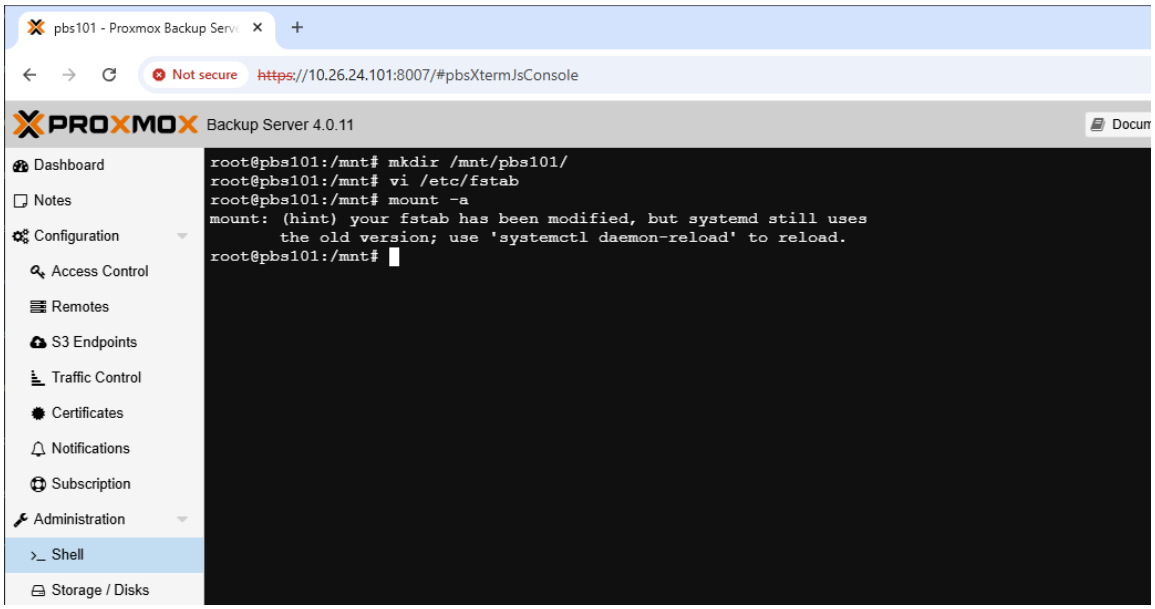
e.g.:



Step 18: Mount fstab listed volumes

Command #1 run: mount -a

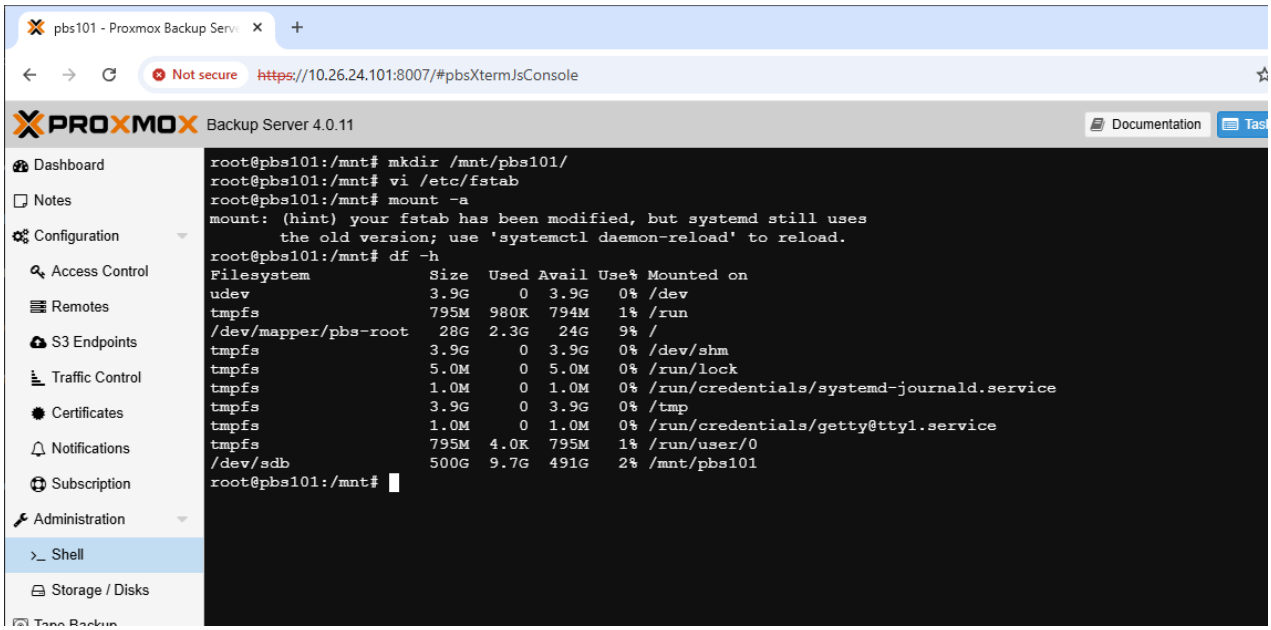
e.g.:



Step 19: Check your work

Command #1 run: running: df -h

e.g.:

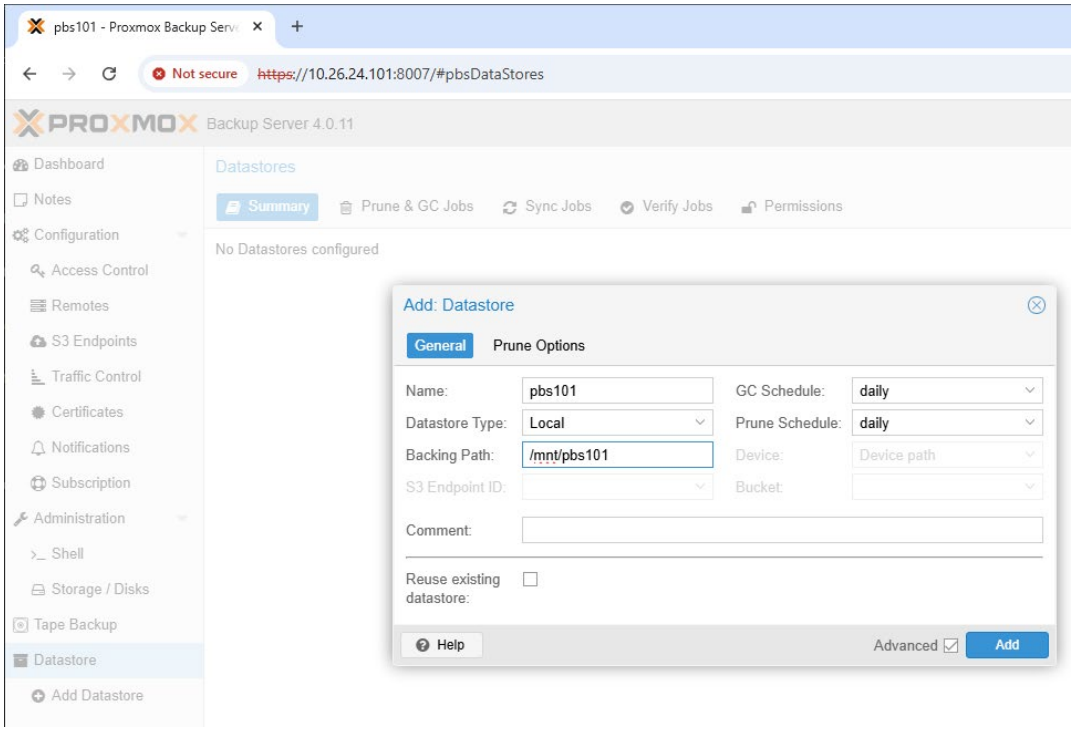


NOTE : You should see /dev/sdb mounted to /mnt/pbsXYZ

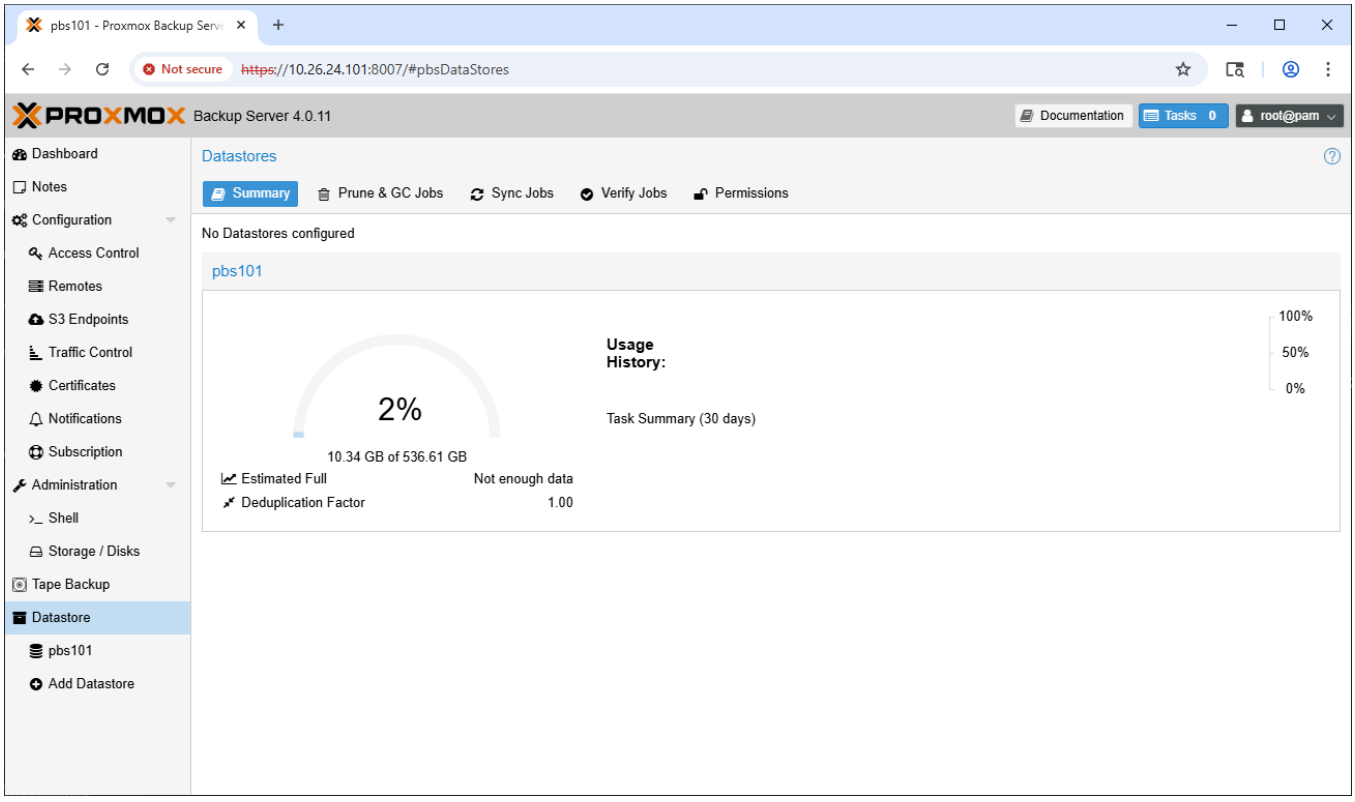
SBS LAB (GUI) – Adding a datastore to PBS

Step 1: pbsXYZ > Datastore > Add Datastore

e.g.:



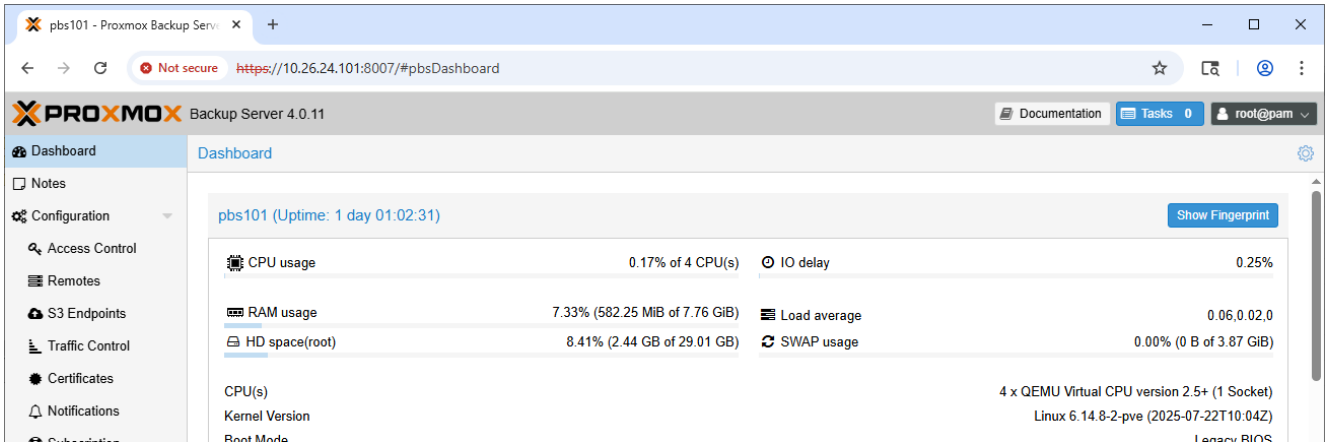
e.g.:



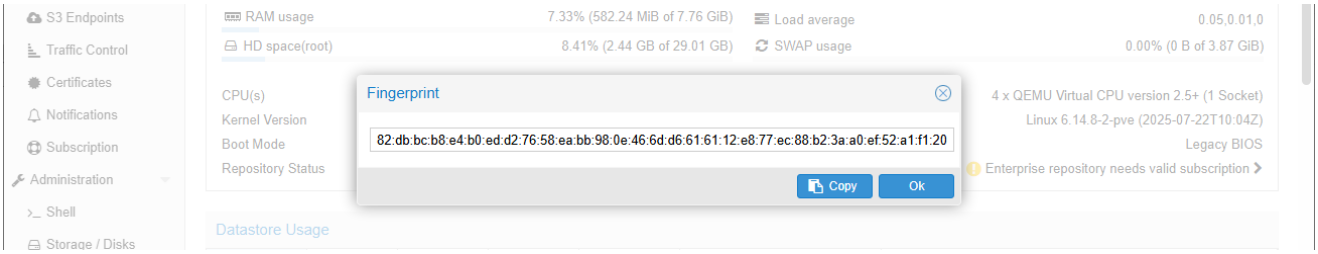
SBS LAB (GUI) – Adding PBS to PVE

Step 1: Goto: pbsXYZ > Dashboard > Show Fingerprint

e.g.:

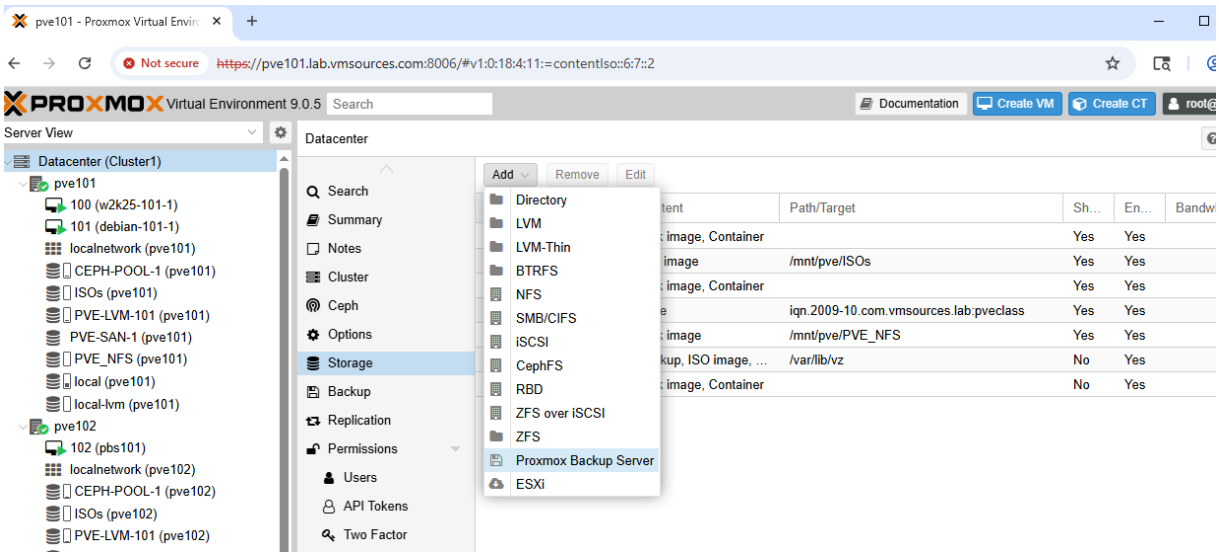


Step 2: Click: Copy



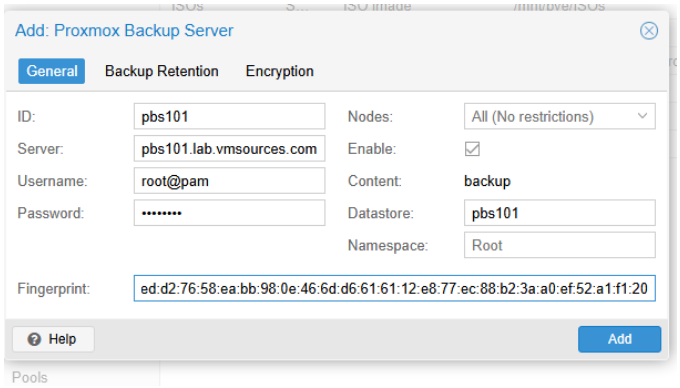
Step 3: Go to PVE Datacenter > Storage > Add > Proxmox Backup Server

e.g.:



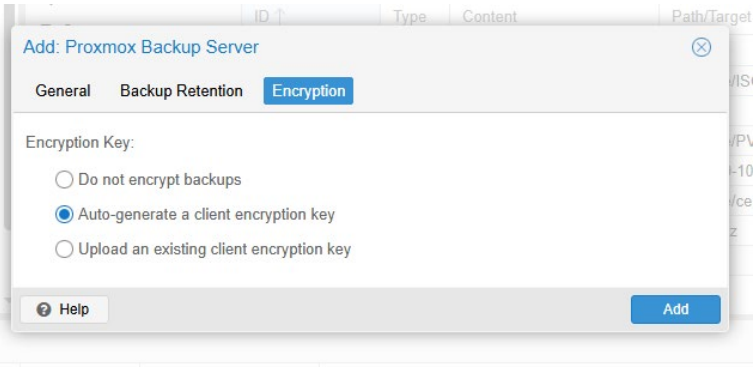
Step 4: Enter the data, substituting your XYZ variable and paste the Fingerprint you copied from pbsXYZ

e.g.:



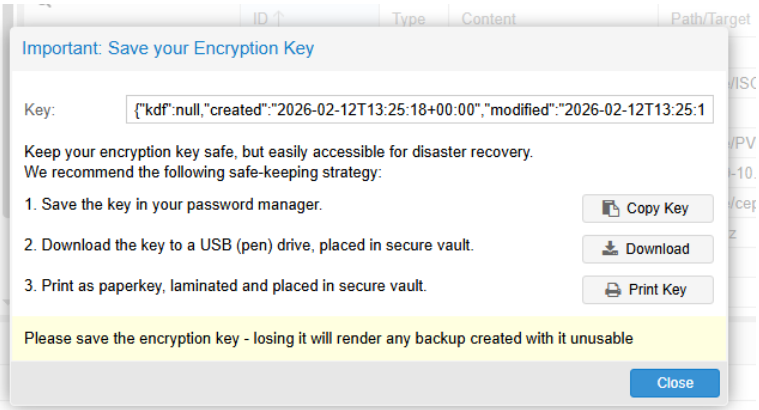
Step 5: Look at Encryption

e.g.:

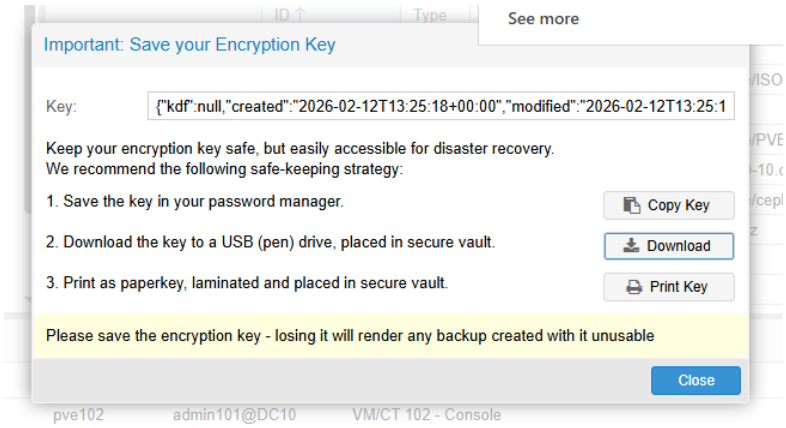


Step 6: Save the key

e.g.:



e.g.:

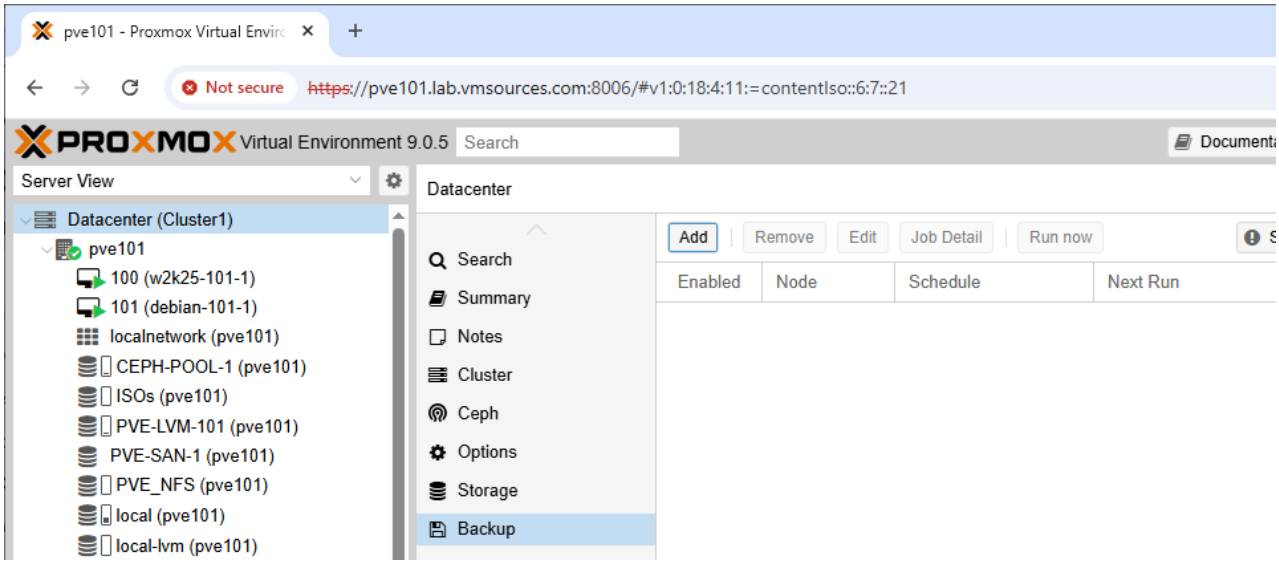


NOTE : Proxmox recommends placing the key in a secure password manager

SBS LAB (GUI) - Running backups to PBS

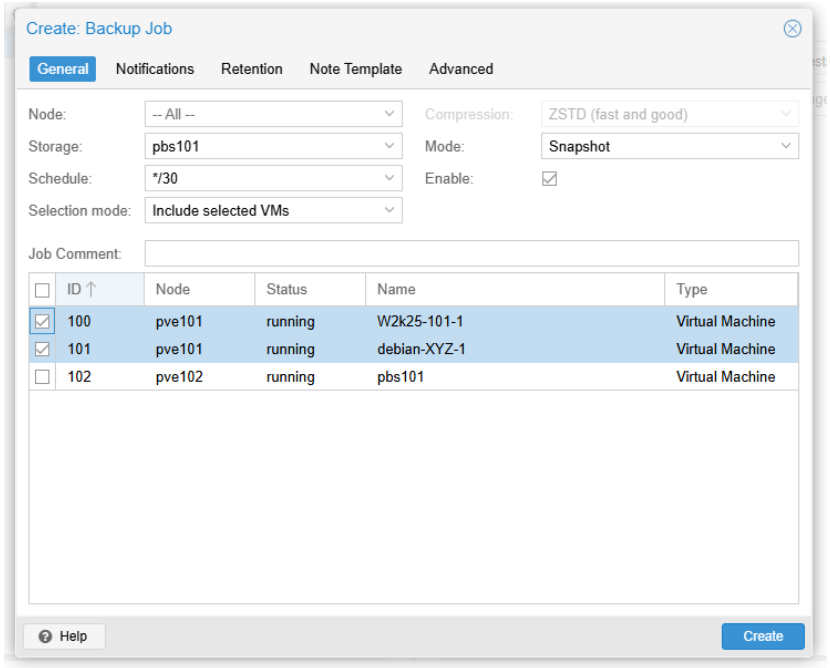
Step 1: Datacenter > Backup > Add

e.g.:



Step 2: Choose your options and add your VMs, excluding the pbsXYZ backup server.

e.g.:

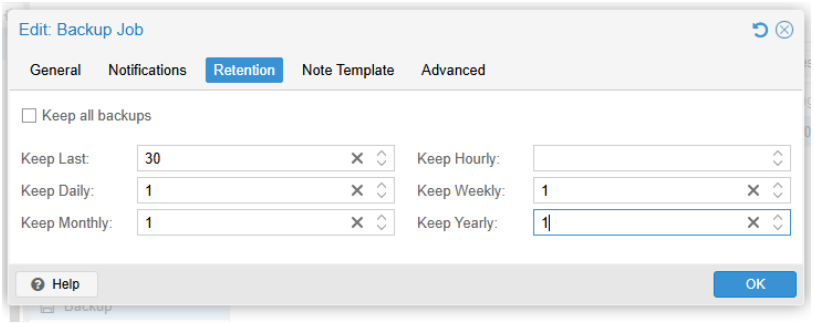


NOTE : We have chosen a 30 minute schedule so we can view results in a short period of time

NOTE : Because we are using an iSCSI mount for backup storage, you actually could backup the PBS backup server – however it is generally not recommended as you might end up backing up all of your backup data and filling the repository.

Step 3: Set retention policy for this job

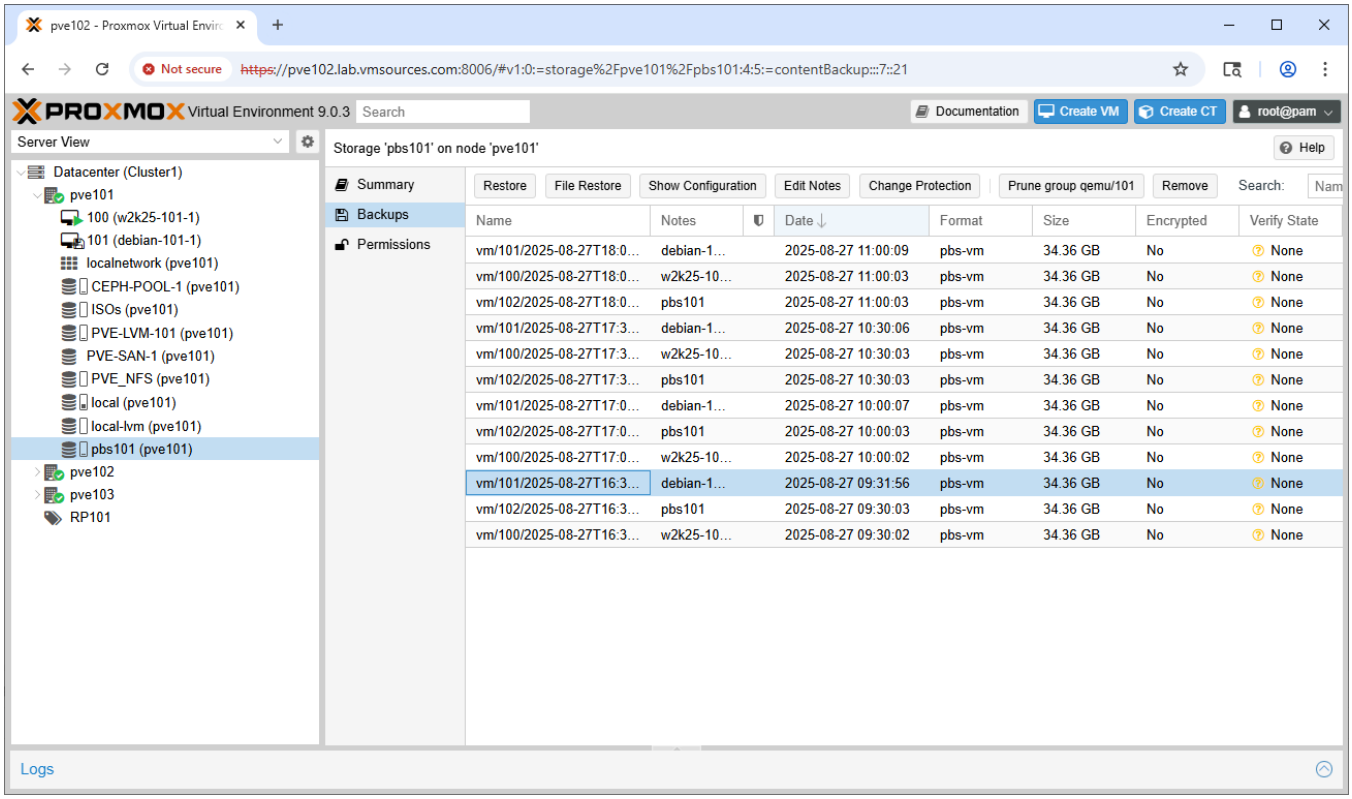
e.g.:



SBS LAB (GUI) – Restores from PBS

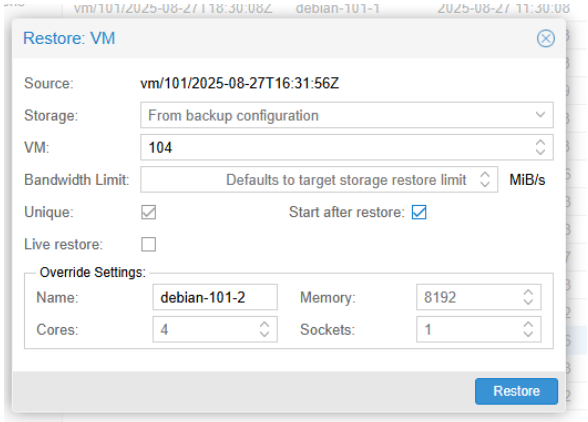
Step 1: pveXYZ > pbsXYZ > Backups > Select VM restore point >Restore

e.g.:

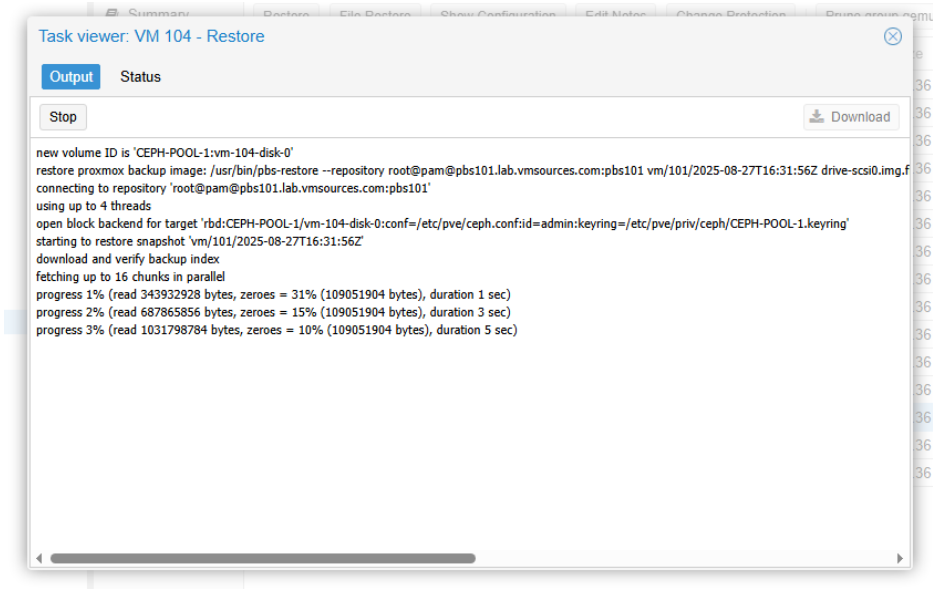


Step 2: Choose your restore options > Click: Restore

e.g.:

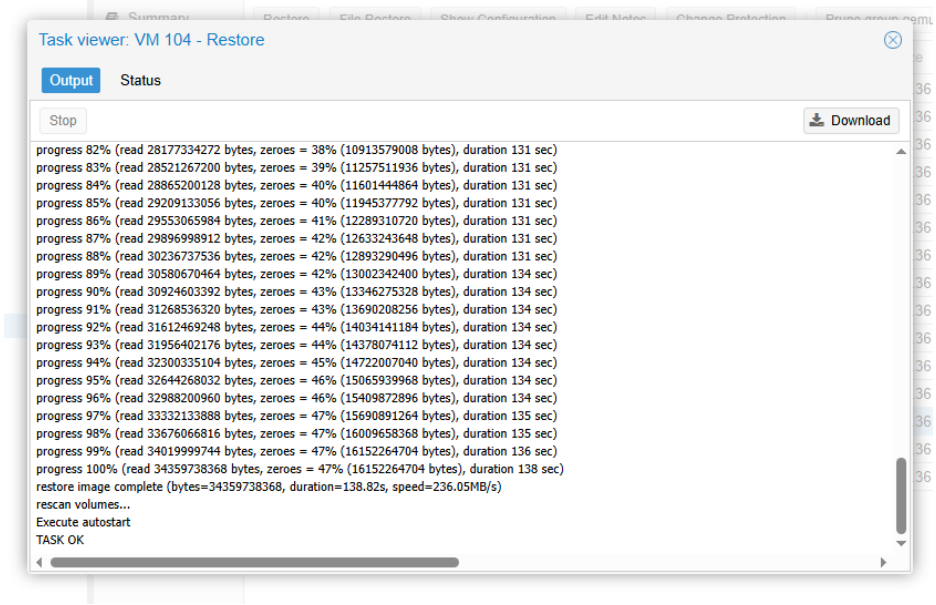


Step 3: Wait



e.g.:

Step 4: Restore completed



e.g.:

Remote Backup and Disaster Recovery

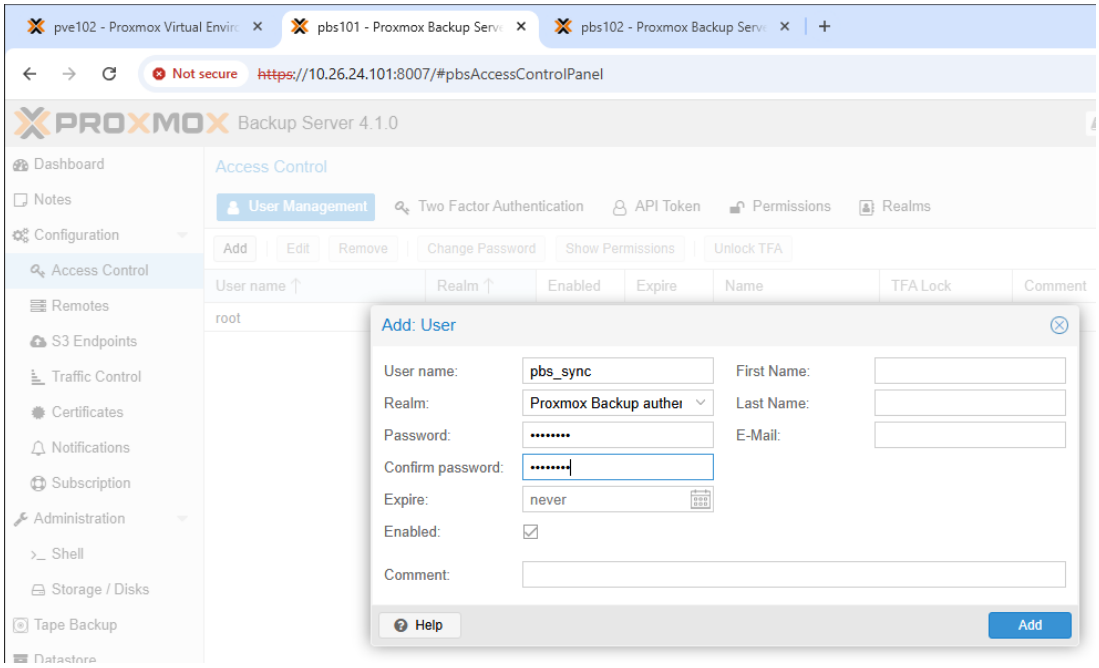
SBS LAB (GUI) – Syncing data between PBS servers

Being able to sync data between two instances of PBS is extremely valuable as it could form the core of a Disaster Recovery plan and/or help complete the 3-2-1 Backup Rule requirements.
The alternate PBS server could be in the same room or on a different continent, so long as network access was possible (VPN, SD WAN).

1. Configure remote user on target server

   Step 1: pbsXYZ > Access Control > User Management > Add

   e.g.:



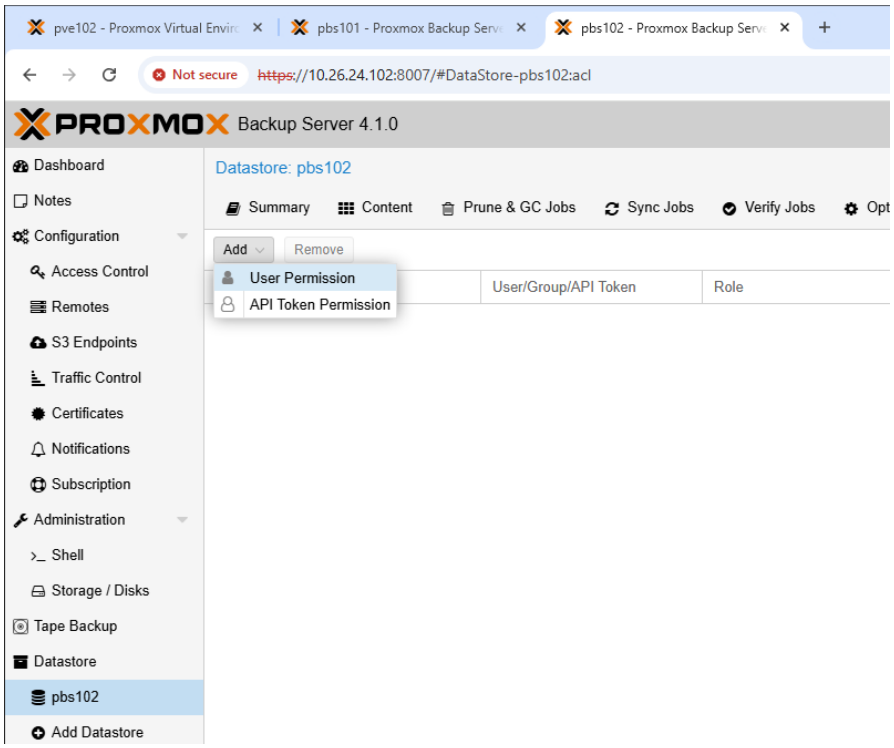
   NOTE : We will do this equally on all pbsXYZ instances so that it is possible to configure replication between servers

   NOTE : This only needs to be done on the target PBS server in production

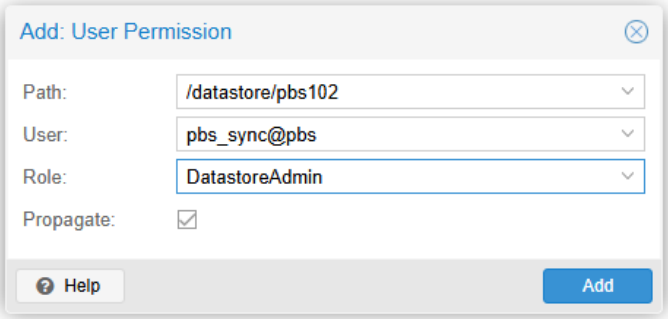
2. Add permissions for pbs_sync user to Datastore

Step 1: pbsXYZ > Datastore > pbsXYZ > Permissions > Add > User Permission

e.g.:



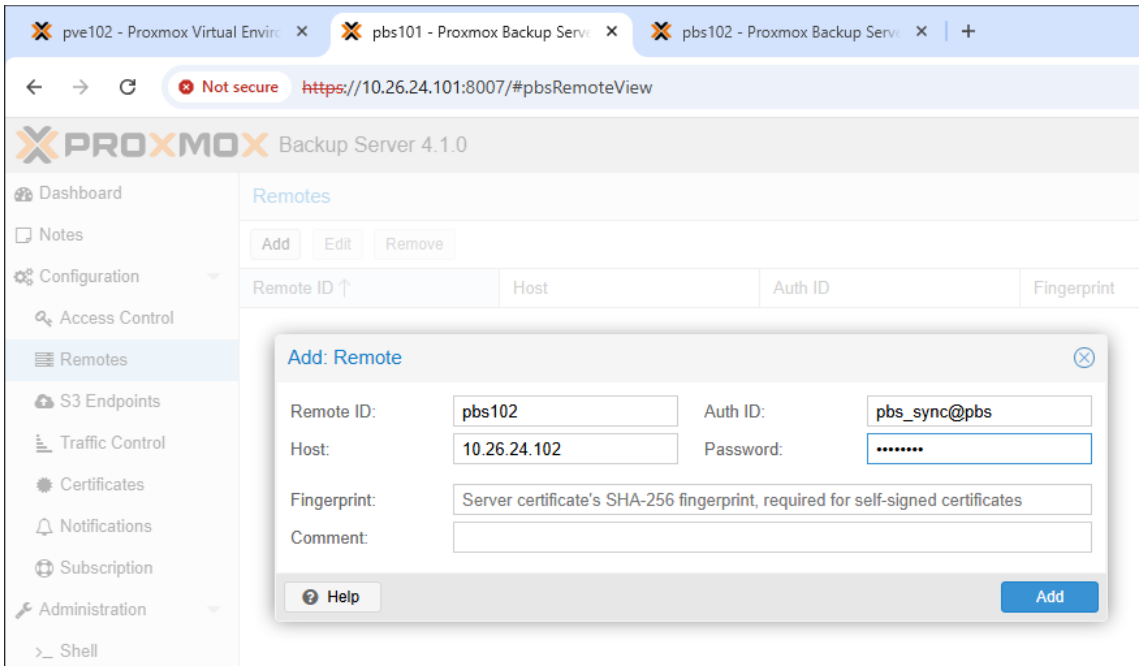
e.g.:



3. Configure Remotes

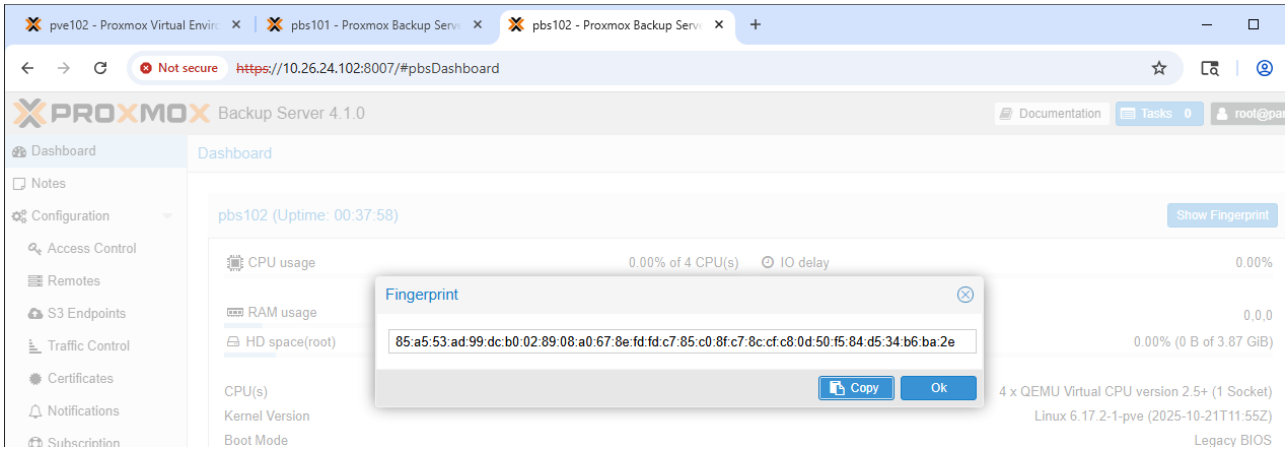
Step 1: pbsXYZ > Remotes > Add

e.g.:



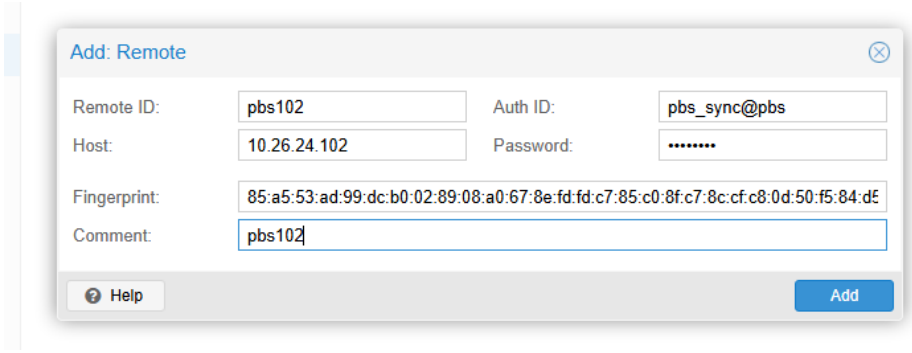
Step 2: Now login to the remote PBS server and copy the fingerprint: pbsREM > Dashboard > Show Fingerprint > Copy

e.g.:



Step 3: Paste the fingerprint and > Add

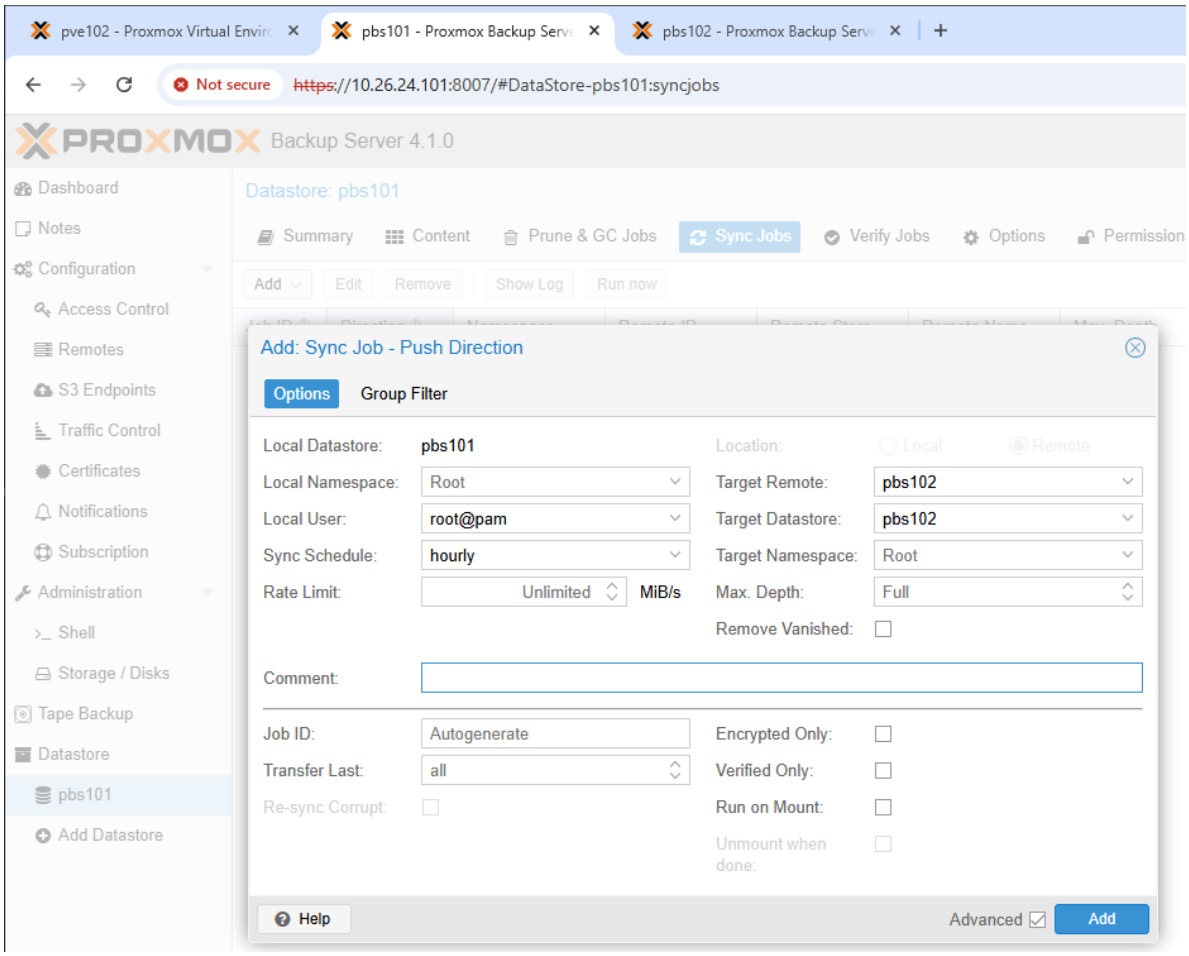
e.g.:



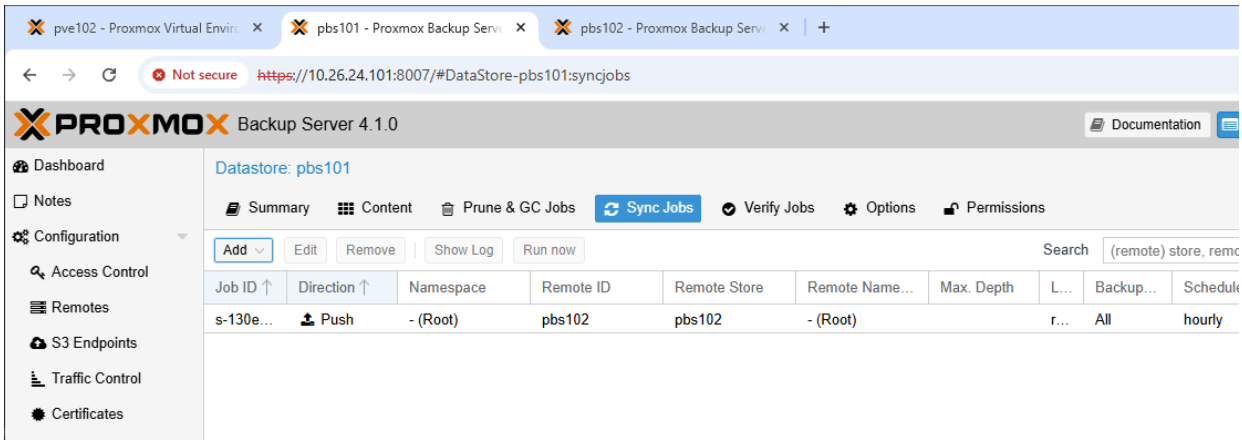
4. Now we need to set up the sync job

Step 1: From the source backup server: pbsXYZ > Datastore > pbsXYZ > Sync Jobs > Add > Add Push Sync Job

e.g.:



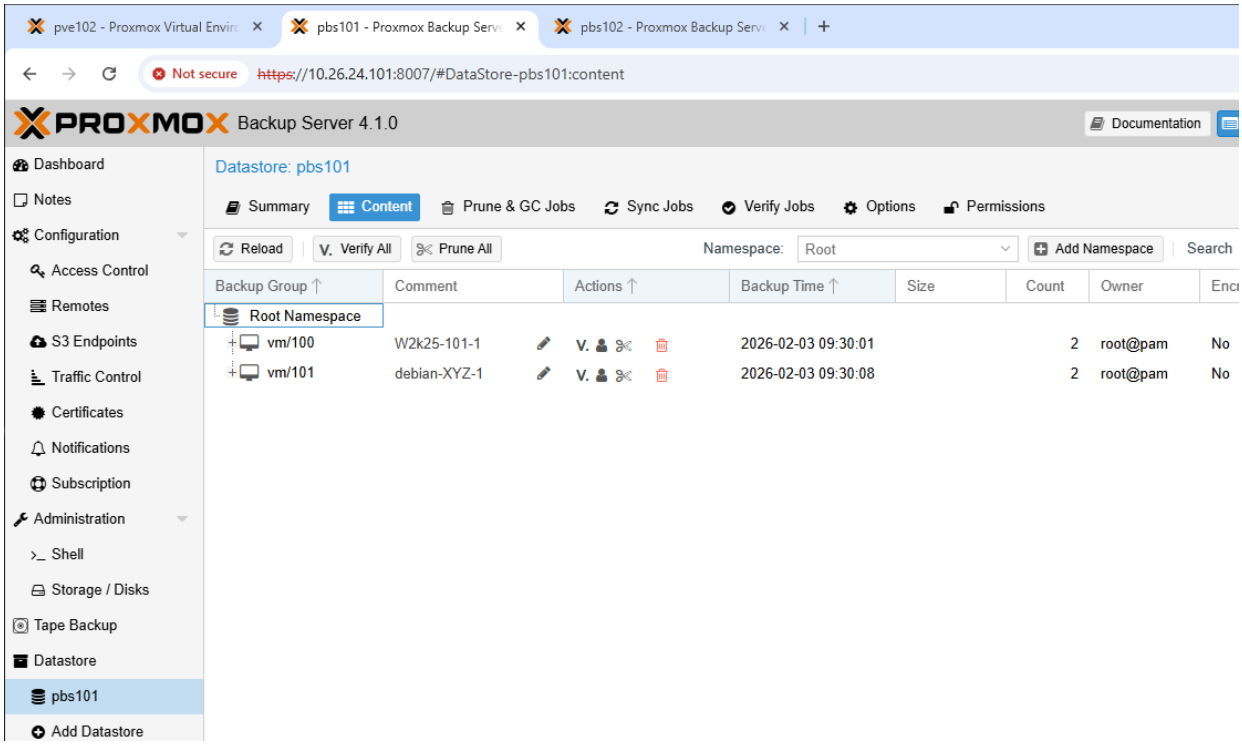
e.g.:



5. Now verify that backups exist:

   Step 1: From the source backup server: pbsXYZ > Datastore > pbsXYZ > Content

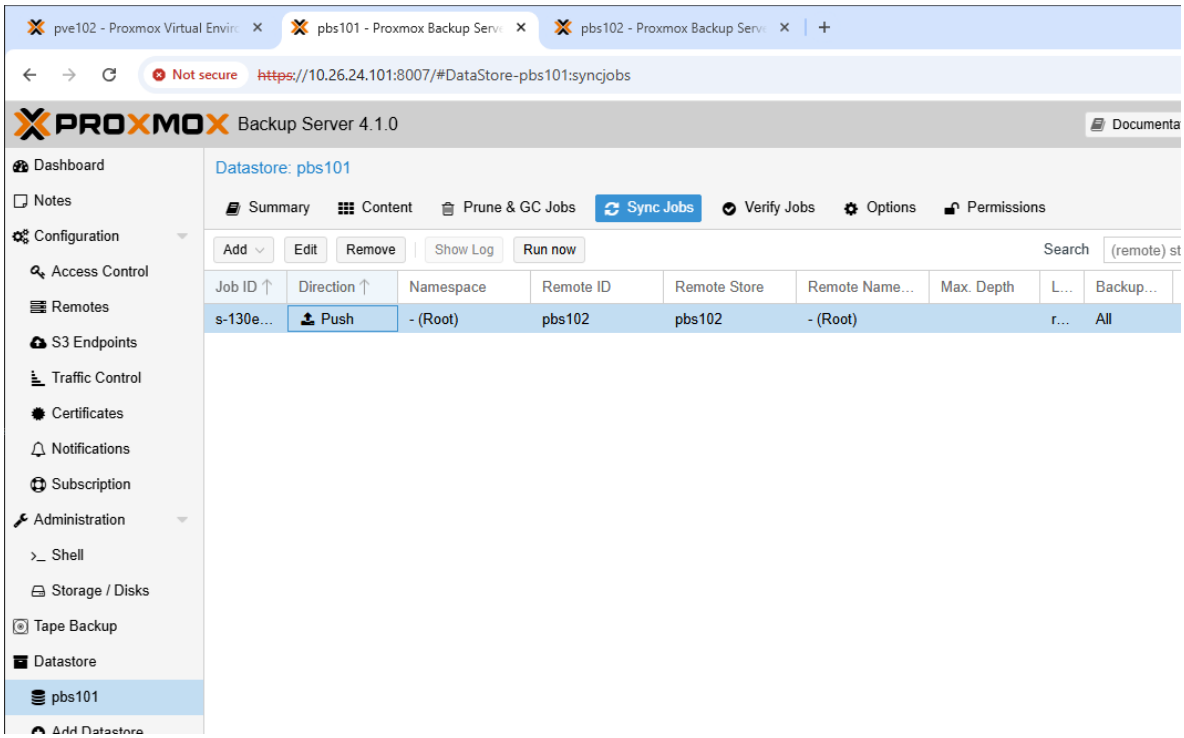
e.g.:



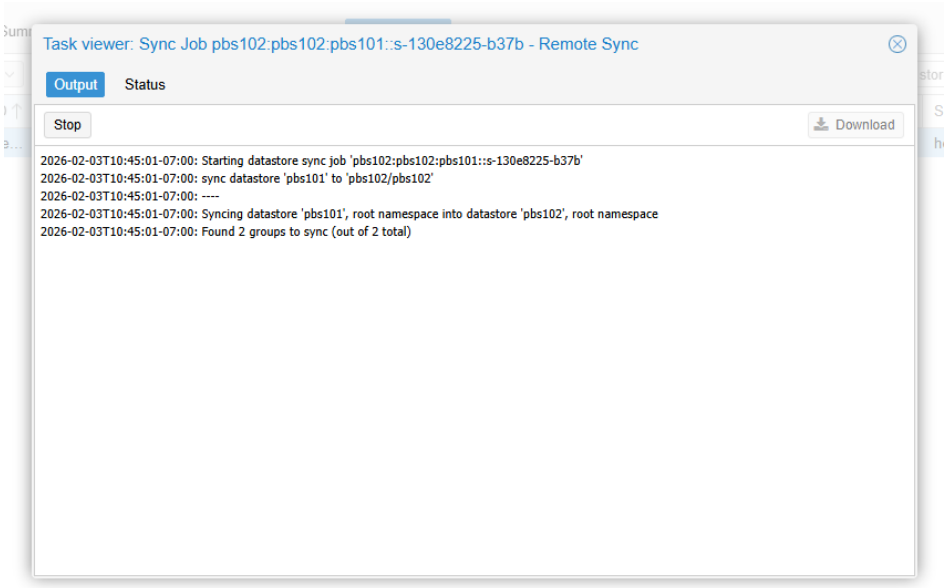
6. Manually run a sync job

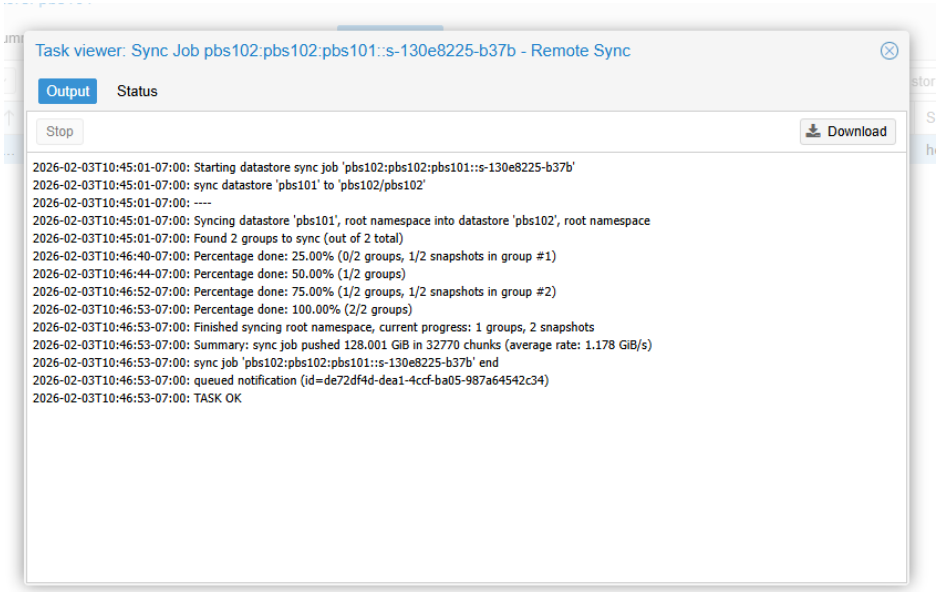
Step 1: From the source backup server: pbsXYZ > Datastore > pbsXYZ > Sync Jobs > Run now

e.g.:



e.g.:

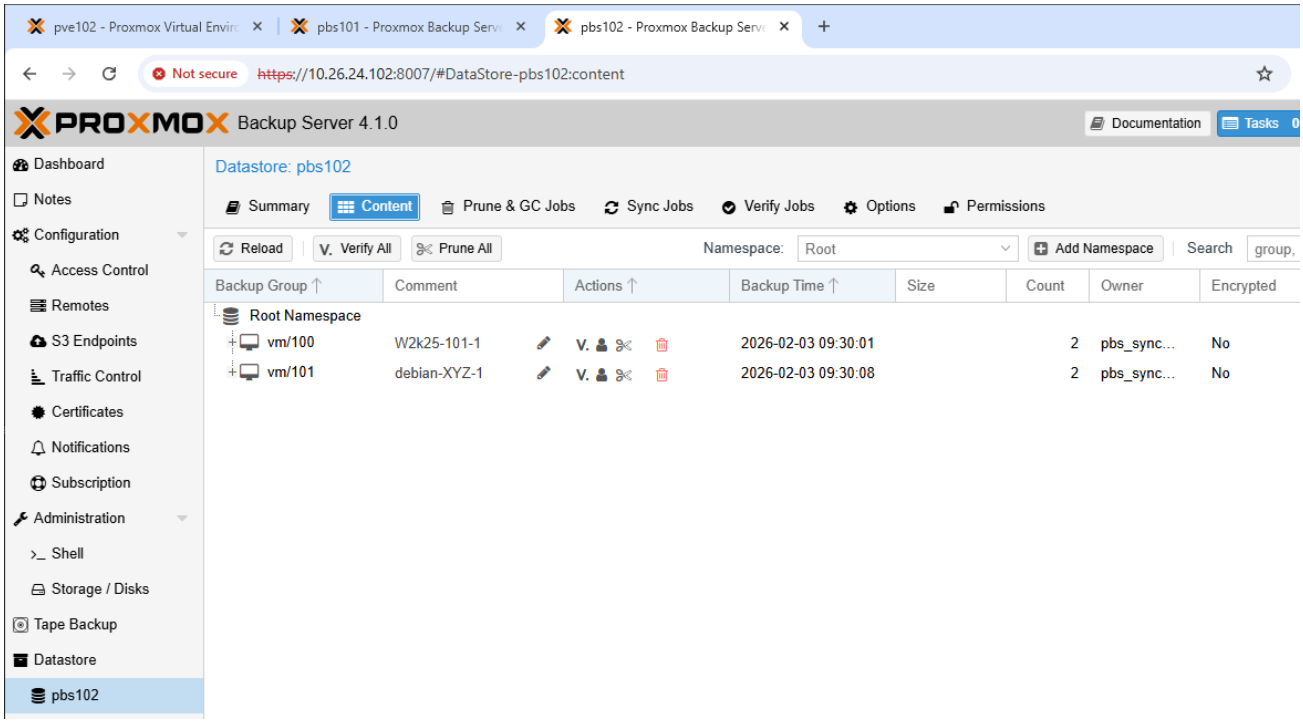




e.g.:

7. Verify data on target PBS

Step 1: From the target backup server: pbsREM > Datastore > pbsREM > Content

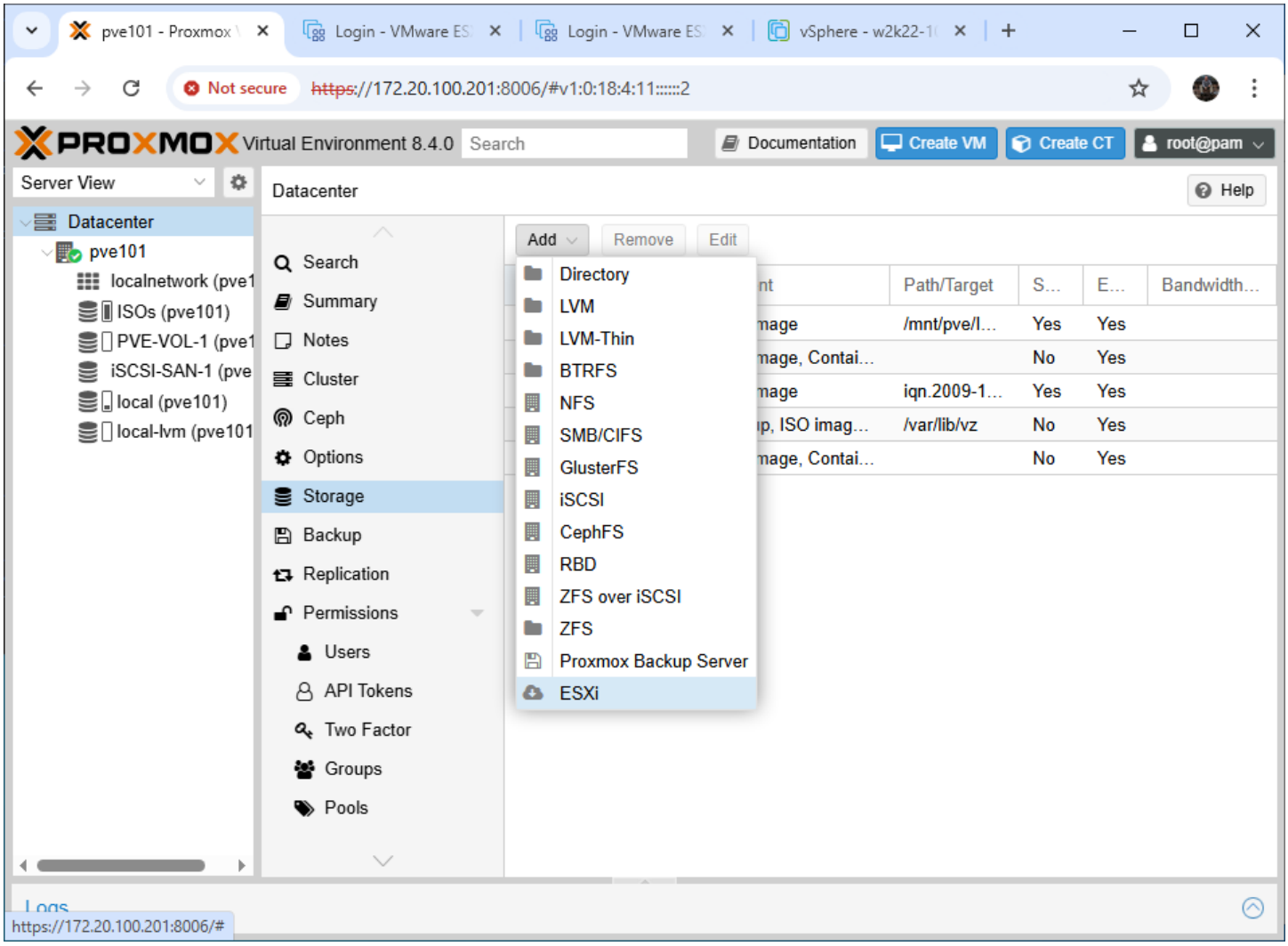


e.g.:

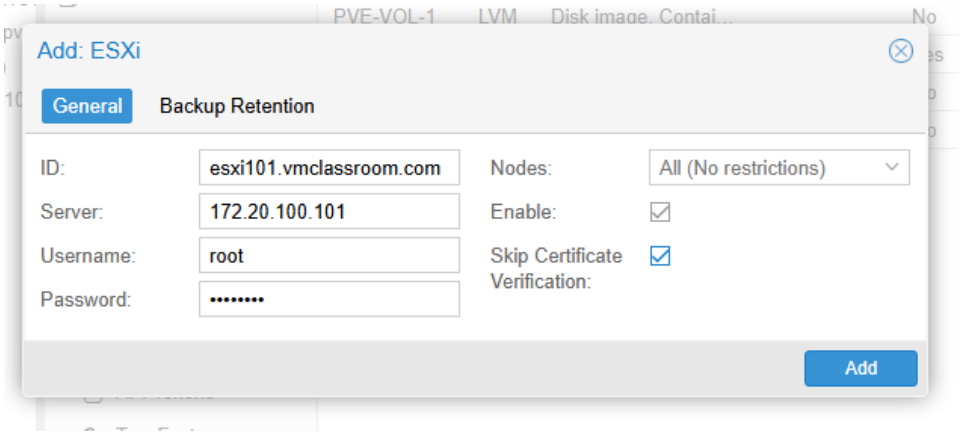
Migrating from another platform to PVE

SBS LAB –(GUI) Migrate VMs from ESXi

1. Now we can add our ESXi host(s): Datacenter > Storage > Add > ESXi

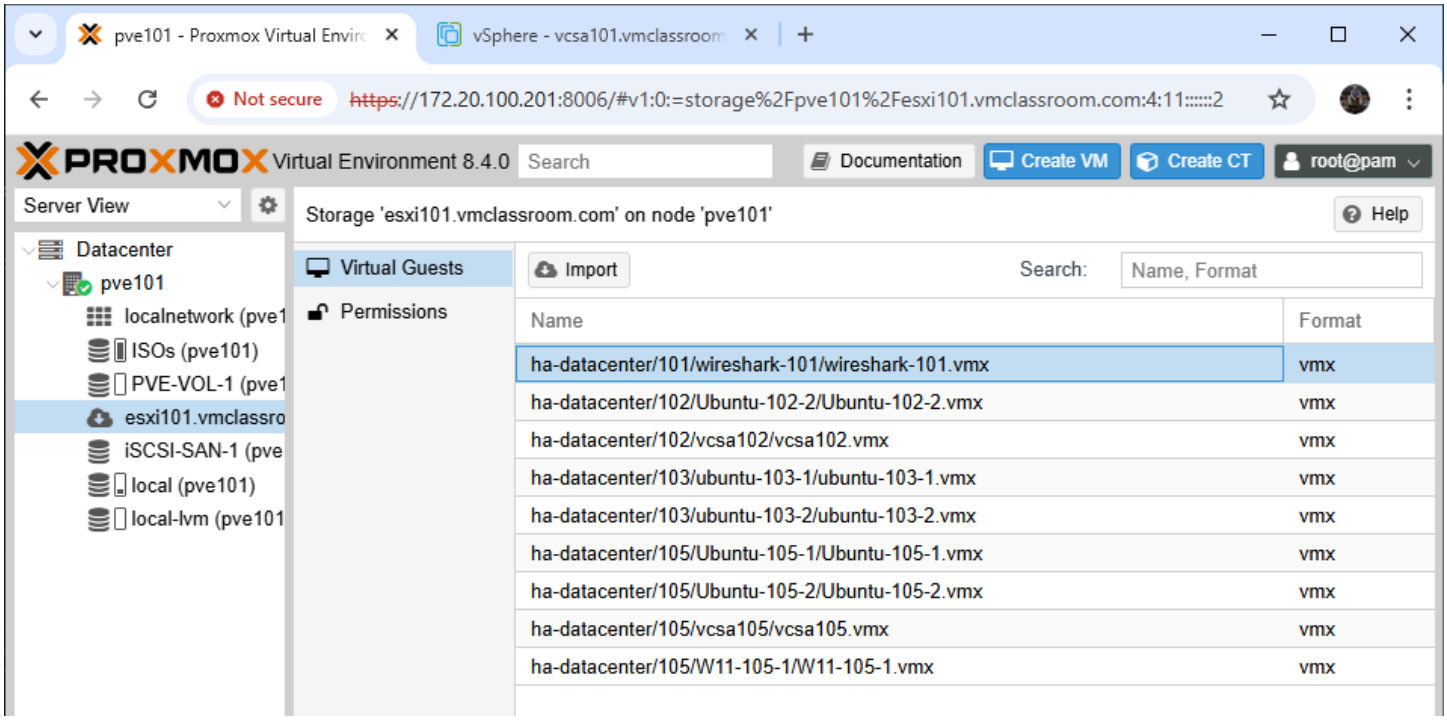


2.

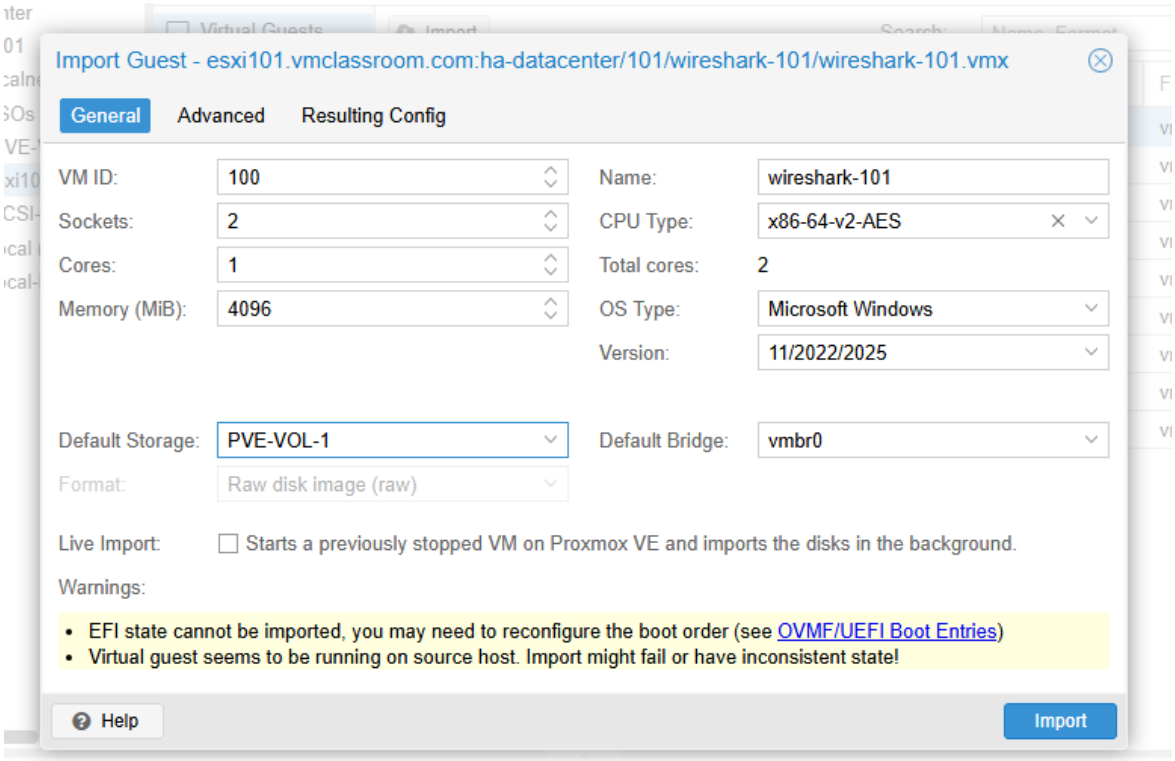


3.

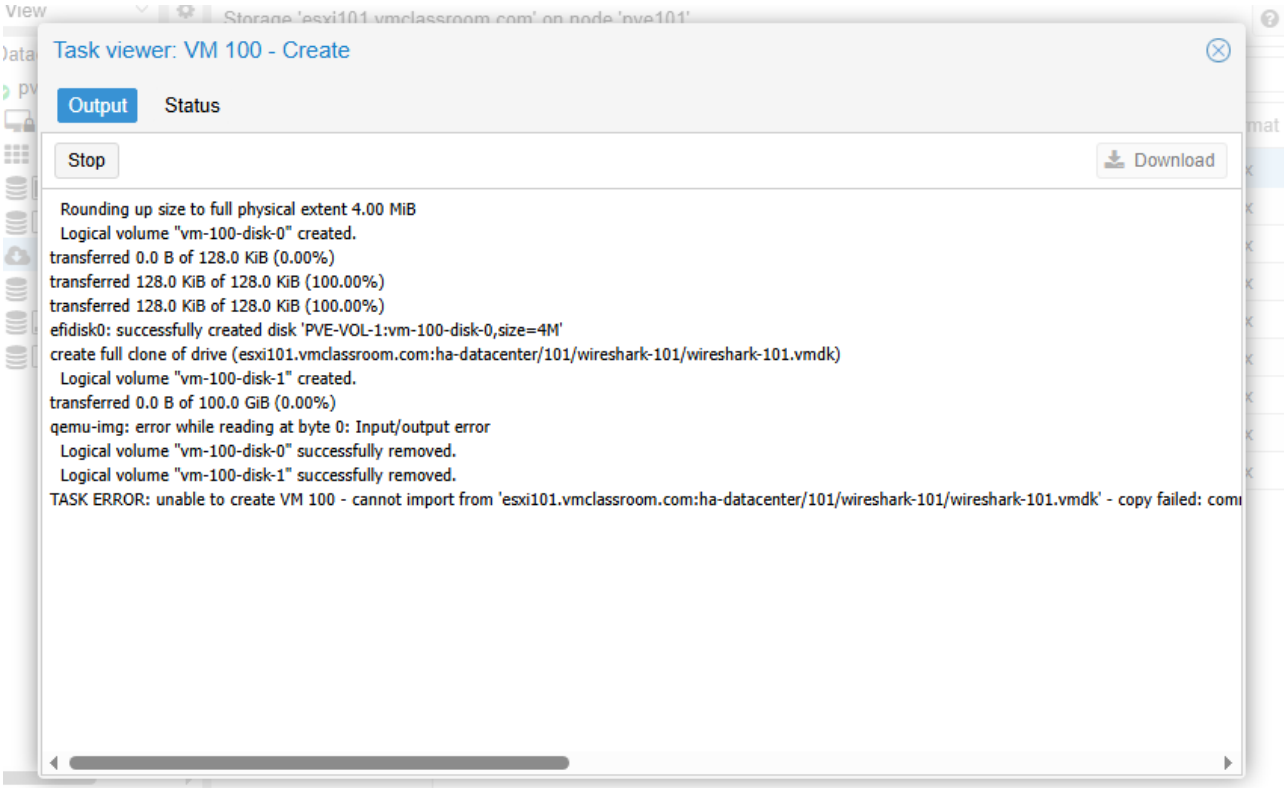
4. At this point we are ready to import VMs from ESXi: PVE Host > ESXi > Import



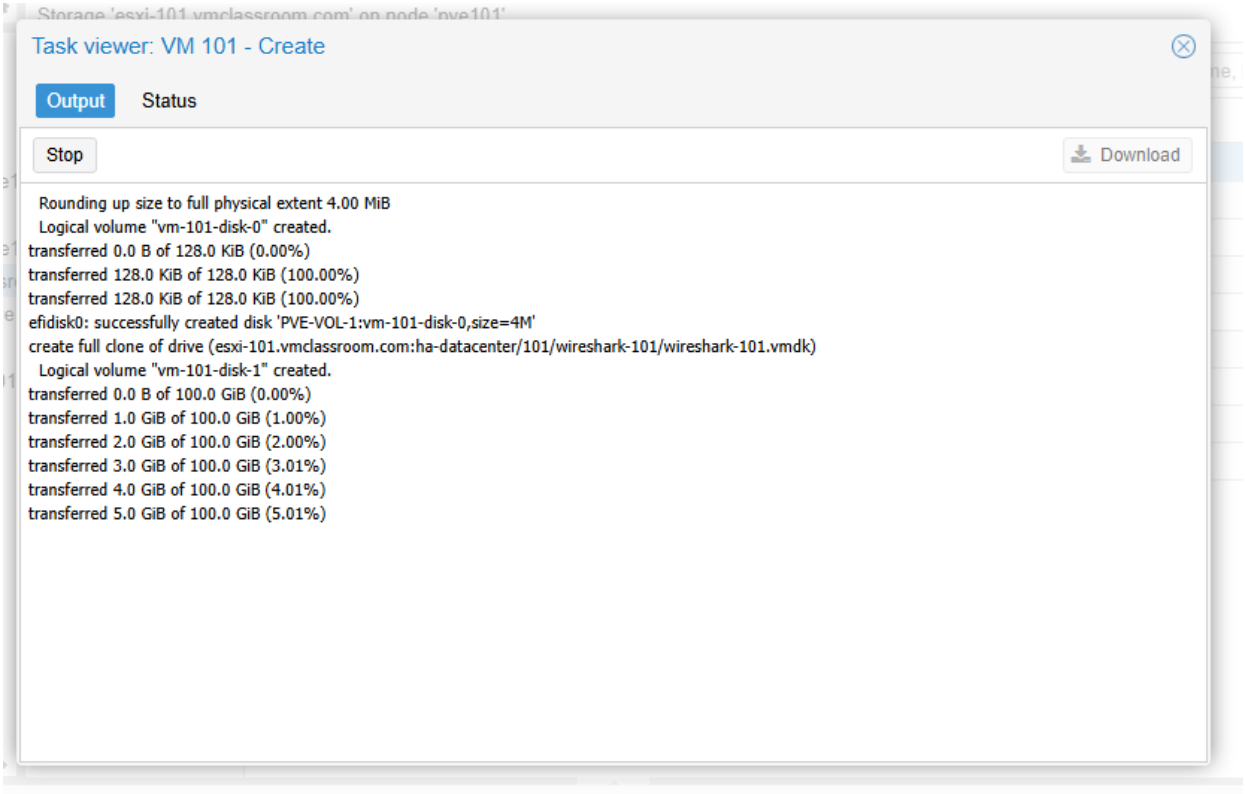
5. Set the parameters, especially destination storage > Import



NOTE : If you get an error, the source VM may be powered-on

6. 

Step 1: This is what it should look like

7. 

Credits / Contributors / Acknowledgements

References

[i] http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf
[ii] http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf
[iii] http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf
[iv] https://en.wikipedia.org/wiki/Classful_network
[v] https://github.com/CumulusNetworks/ifupdown2/issues/124
[vi] https://pve.proxmox.com/wiki/Network_Configuration
[vii] https://pve.proxmox.com/wiki/ISCSI_Multipath
[viii] https://pve.proxmox.com/wiki/ISCSI_Multipath
[ix] https://pve.proxmox.com/wiki/ISCSI_Multipath