Implementing Dell PowerFlex with VMware Tanzu

August 2023

H19672

Implementation Guide

Abstract

This document provides guidance for implementing Dell PowerFlex with VMware Tanzu and TKG Clusters using both SCSI and NVMe/TCP connectivity.

Dell Technologies PowerFlex Engineering

PowerFlex Engineering
Validated

DELL Technologies

Copyright

Contents

Chapter 1 Introduction

This chapter presents the following topics:

Executive summary

Containers are a way of packaging software so that it can be run reliably and efficiently on any computing environment. A container image is a lightweight, stand-alone, executable package of software that includes everything that is required to run an application: code, runtime, system tools, system libraries, and settings.

Kubernetes is an open-source container orchestration system that automates the deployment, scaling, and management of containers and containerized applications. Kubernetes has a toolset and an API to aid in this management across all the hosts in the environment. Kubernetes handles the scaling, resiliency, scheduling, and load balancing of the containers. Kubernetes groups containers into pods.

vSphere with Tanzu (Tanzu) provides an integrated solution for running Kubernetes workloads natively on the ESXi hypervisor layer. Through the vCenter, VMware simplifies the creation of Kubernetes clusters and gives both administrators and developers an interface to work with containers. Dell PowerFlex offers both block and file storage, the types consumed by vSphere with Tanzu. The Administrators can control the number of resources available to developers, including storage, while developers can manage the entire life cycle of their Tanzu Kubernetes clusters.

PowerFlex is a software-defined storage platform that is designed to reduce operational and infrastructure complexity. It empowers organizations to move faster by delivering flexibility, elasticity, and simplicity with predictable performance and resiliency at scale. PowerFlex 4 supports both block and file, the types of storage consumed by vSphere with Tanzu. In addition, Tanzu can consume block storage through both SCSI (SDC) and NVMe/TCP.

Document purpose

This guide provides configuration details for implementing vSphere with Tanzu on PowerFlex storage. It demonstrates how to use both SCSI and NVMe/TCP connectivity to create persistent volumes on PowerFlex for Kubernetes environments running on VMware.

Note: Examples that are provided in this guide cover performing various VMware vSphere activities using PowerFlex systems and Dell software. These examples were developed for laboratory testing and may need tailoring to suit other operational environments. Any procedures that are outlined in this guide should be thoroughly tested before implementing in a production environment.

Note: The contents of this document are valid for the described software and hardware versions. For information about updated configurations for newer software and hardware versions, contact your Dell Technologies sales representative.

Audience

This document is intended for use by storage administrators, system administrators, and VMware vSphere administrators.

Readers of this document are expected to be familiar with the following topics:

- Dell PowerFlex system operation
- Dell software including the following: PowerFlex Manager (4.x)
- Containers and in particular Kubernetes
- VMware vSphere products and their operations, including Tanzu

Table 1. Revision history

Part number	Release date	Description
H19672	August 2023	Initial release

Wizard walkthroughs

Users are expected to be familiar with the products and technologies that are covered in the guide. Not every step of each wizard has an associated image. The steps are always documented.

Terminology

The following table provides definitions for some of the terms that are used in this document.

Table 2. Terminology

Term	Definition
ALB	NSX Advanced Load Balancer
ITOM	IT Operational Management
LCM	Life Cycle Management
MDM	Meta Data Manager
SDC	Storage Data Client
SDS	Storage Data Server
SDT	Storage Data Target (TCP)
VM	Virtual Machine
K8S	Kubernetes
LCM	Lifecycle management including Day 0, Day 1, and Day 2 operations
TKGS	Tanzu Kubernetes Grid Service aka vSphere with Tanzu

Term	Definition
Namespaces	vSphere Namespace, introduced in vSphere 7, creates multitenancy. vSphere Namespace is a vSphere concept to provide multitenancy, and separation of resources belongs to one particular tenant. When a Namespace is created on a Supervisor Cluster, that namespace is created in the cluster itself (`kubectl get namespaces`).
TKG Cluster	Tanzu Kubernetes Grid Cluster, an upstream K8S cluster created for DevOps workloads using the TKG Service
TKC	The synonym for TKG Cluster, stands for Tanzu Kubernetes Cluster.
Guest Cluster	The synonym for TKGS Cluster, a term to indicate that the cluster is outside of vSphere primitives, and life cycle management is independent of vSphere LCM.
VDS	vSphere Distributed Switch (defined and managed by vCenter)
VSI	Dell Virtual Storage Integrator

Chapter 2 Product overview

This chapter presents the following topics:

PowerFlex family overview

PowerFlex software-defined infrastructure enables broad consolidation across the data center, encompassing almost any type of workload and architecture. The software-defined architecture offers automation and programmability of the complete infrastructure and provides scalability, performance, and resiliency to enable effortless adherence to stringent workload SLAs.

The PowerFlex family provides a foundation that combines compute and high-performance storage resources in a managed unified fabric. PowerFlex comes in flexible deployment options (rack, appliance, or custom nodes and in the public cloud) that enables independent (two-layer), HCI (single-layer), or mixed architectures. PowerFlex is ideal for high-performance applications and databases, building an agile private-hybrid cloud, or consolidating resources in heterogeneous environments.

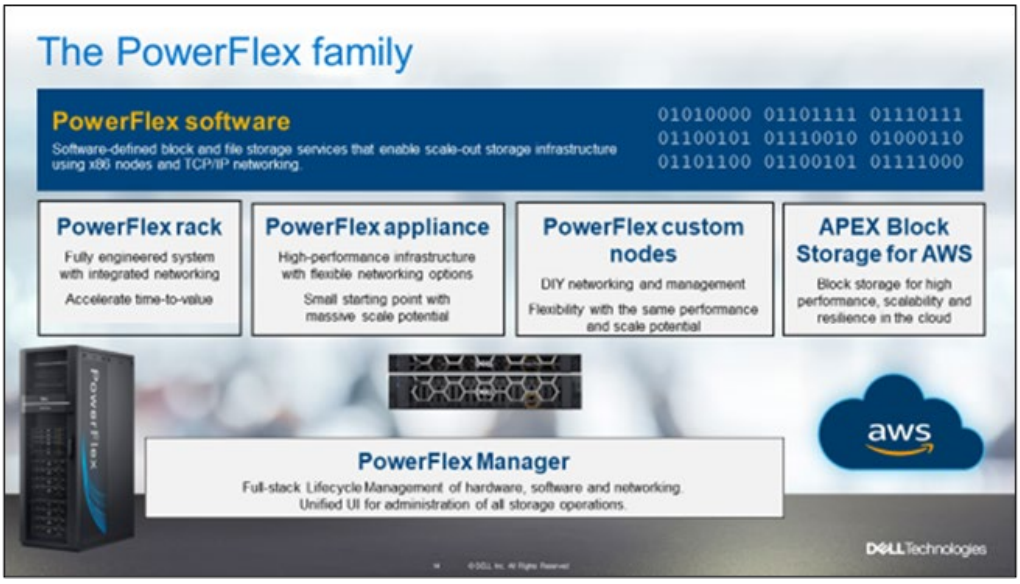

Figure 1. PowerFlex family

PowerFlex software components

Software is the key differentiation in the PowerFlex offering. PowerFlex software components not only provide software-defined storage services, but also help simplify infrastructure management and orchestration. This software enables comprehensive IT Operational Management (ITOM) and Life Cycle Management (LCM). These capabilities span compute and storage infrastructure, from BIOS and Firmware to nodes, software, and networking.

PowerFlex

PowerFlex is the software foundation of PowerFlex software-defined infrastructure. It is a scale-out block and file storage service that is designed to deliver flexibility, elasticity, and simplicity with predictable high performance and resiliency at scale.

PowerFlex Manager

PowerFlex Manager is the software component in the PowerFlex family that enables ITOM automation and LCM capabilities for PowerFlex systems. Starting with PowerFlex 4.0, the unified PowerFlex Manager brings together three separate components from previous releases: PowerFlex Manager, the core PowerFlex UI, and the PowerFlex

gateway. The new PowerFlex UI runs in Kubernetes and embraces a modern development framework.

PowerFlex file services

PowerFlex File Controllers, also known as File Nodes, are physical nodes that enable PowerFlex software defined File Services. They host the NAS Servers, which in turn host tenant namespaces and file systems, mapping PowerFlex volumes to the file systems presented by the NAS Servers. All major protocols are supported, such as NFS, SMB-CIFS, FTP, and NDMP. PowerFlex file service is supported from PowerFlex 4.0.

PowerFlex CSI and CSM

An important component outside of PowerFlex that enables a flexible consumption model for Kubernetes is the PowerFlex CSI driver. It was developed as a part of the Dell Kubernetes strategy. After the CSI driver for PowerFlex is loaded into Kubernetes, it can be used to provision persistent volumes from the underlying PowerFlex storage resource. If the Kubernetes deployment is running low on PowerFlex storage resources, you can add PowerFlex storage nodes to increase the system capacity and performance.

The CSI driver connects the PowerFlex system and Kubernetes deployments. It is a storage broker agent which dynamically provisions volumes from PowerFlex through the PowerFlex API gateway to the Kubernetes cluster. Once the volume is available on PowerFlex, it is immediately mapped to the requesting pod. If a pod is destroyed or rescheduled, the CSI plug-in ensures that the volumes are remapped upon rescheduling of that pod.

Customers running Kubernetes clusters on PowerFlex use the Dell Container Storage Modules (CSM), which extend the CSI driver capabilities. These modules:

- Provide enterprise storage capabilities to Kubernetes for cloud-native stateful applications.
- Reduce management complexity, so that developers can independently use enterprise storage with ease and automate daily operations.
- Extend storage functionality and capabilities beyond using the CSI driver alone.

These modules include snapshot, observability, authorization, application mobility, and resiliency.

PowerFlex supports multiple operating systems and different deployment options for on-premises and public cloud deployment models (available in AWS). PowerFlex is validated with the leading Kubernetes distributions as shown in Figure 2.

Figure 2. PowerFlex for different Kubernetes distributions

PowerFlex deployment architectures

PowerFlex software-defined infrastructure excels in deployment flexibility. PowerFlex can be deployed in a two-layer (independent compute and storage layers), single-layer (Hyperconverged Infrastructure, or HCI), or a mixture of the two architectures (Mixed).

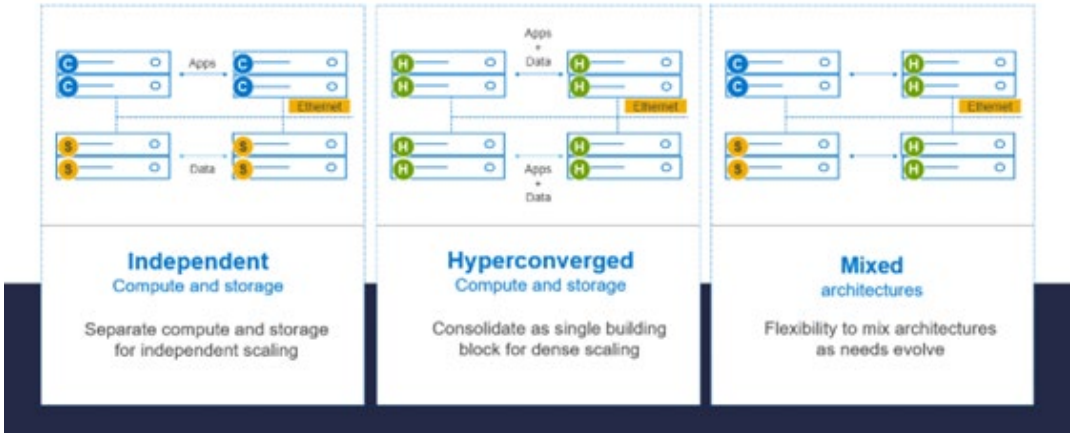

Figure 3. PowerFlex architectures

Independent architecture

In an independent architecture or two-layer architecture, some nodes provide storage capacity for data in applications. Other separate and independent nodes provide compute resources for applications and workloads. Compute and storage resources can be scaled independently by adding nodes to the cluster while it remains active. This separation of compute and storage resources helps to minimize software licensing costs in certain situations. This architecture can be ideal for high-performance databases and application workloads.

Hyperconverged architecture

In an HCI architecture, each node in the cluster contributes storage and compute resources simultaneously to the applications and workloads. This architecture allows you to scale your infrastructure uniformly with building blocks that add both storage and

compute resources. This architecture is appropriate for data center and workload consolidation.

Mixed architecture

A mixed architecture has a combination of both the HCI and Independent architectures. As shown in Figure 3 above, there would be some storage only nodes, compute only nodes, and hyperconverged nodes as part of the same PowerFlex cluster. This architecture is desirable when working with an existing compute infrastructure and adding high-performance software-defined infrastructure. This architecture can also be a starting point for a two-layer deployment design as external workloads are migrated to PowerFlex.

PowerFlex consumption options

PowerFlex rack

PowerFlex rack is a software-defined infrastructure platform that delivers flexibility, elasticity, and simplicity with predictable performance and resiliency at scale. It combines compute and high-performance storage resources in a managed unified network. This rack-based engineered system, with integrated networking, enables customers to achieve the scalability and management requirements of a modern data center.

PowerFlex appliance

PowerFlex appliance is a PowerEdge server which has been configured to be a node in a software-defined infrastructure deployment that runs PowerFlex software components. This offering allows customers the flexibility and savings to bring their own compatible networking.

PowerFlex Custom Nodes

PowerFlex Custom Nodes are validated server building-blocks configured for use with PowerFlex. They are available with thousands of configuration options and are available for customers who prefer to build their own environments.

Dell APEX Block Storage for AWS

PowerFlex software can also be deployed in the public cloud. It is available in the Amazon Marketplace as Dell APEX Block Storage for AWS (formerly known as PowerFlex cloud storage). PowerFlex on AWS offers the same on-premises benefits of high-performance, linear scalability, and high resilience as in cloud. PowerFlex also adds cloud-specific benefits, such as large volume sizes, extreme performance based on NVMe drives, and predictable scalability. With PowerFlex on AWS, you can also get higher Multi Availability Zone (Multi AZ) resiliency when PowerFlex Fault sets are distributed across multiple AWS Availability Zones. For more information, see Dell APEX Block Storage for AWS.

Tanzu

vSphere with Tanzu, or Tanzu, enables the running of Kubernetes workloads natively on the hypervisor layer. VMware makes this functionality possible through a feature that is called Workload Management which is part of vCenter. The essential component of Workload Management is the Supervisor Cluster, which is a Kubernetes implementation. Users can add this cluster to an existing, supported vSphere cluster. Users can then create dedicated resource pools (namespaces) for Tanzu Kubernetes clusters which are run directly on the ESXi hosts on virtual machines. Users can then deploy applications on those clusters.

In order to use vSphere with Tanzu, one of two networking options is required:

- NSX-T
- vSphere Networking (VDS)

When using vSphere Networking, a load balancer is necessary. There are two load balancers to choose from:

- NSX Advanced Load Balancer
- HAProxy Load Balancer

As NSX-T is a complete networking solution, it enables the use of two features that vSphere Networking does not: vSphere Pods and a Harbor Registry. The environment that is detailed in this guide uses vSphere Networking as it is a simpler and less costly implementation.

Note: PowerFlex does not support VMware Virtual Volumes with PowerFlex 4.0.x, so it is not an available option for Tanzu storage.

Note: PowerFlex does not support vSphere 8 with PowerFlex 4.0.x, so vSphere 7.0.3 is used in this configuration.

Chapter 3 Environment and system requirements

This chapter presents the following topics:

Introduction

The following sections cover the requirements for implementing VMware vSphere with Tanzu on a PowerFlex 4.0.x.

Prerequisites

The following prerequisites must be met before implementing vSphere with Tanzu on PowerFlex. They are meant to be high level as some tasks related to them are covered in the guide.

- VMware vCenter 7.0.3 must be installed and configured according to the vSphere Tanzu documentation. This configuration includes enabling Dynamic Resource Scheduler (DRS) and High Availability (HA) on the ESXi cluster that is used with Tanzu.
  - If using VSI for SDC installation and datastore creation, it must be deployed and registered in the vCenter.
- The NSX Advanced Load Balancer must be installed and configured per the Tanzu documentation.
- A PowerFlex system must be installed and configured with storage available for provisioning. Although both SCSI and NVMe/TCP are used in this solution, only one type of host connectivity to vSphere is required which could also be NFS. The type of PowerFlex system (for example, appliance, custom node) is not critical to the solution.
- Licenses must be obtained for PowerFlex, NSX Advanced Load Balancer, and VMware (vCenter and ESXi).

Solution architecture

This section describes the different types of topologies that make up the solution.

Network topology

At the physical layer, two supported Top of Rack (TOR) switches are used for redundancy and load-balancing purposes. There are four connections from each Dell host, two to TOR A and two to TOR B. Three of the connections are used, one for PowerFlex management and two for PowerFlex data. The data connections are split between the switches.

The following table shows the different networks that are configured for this solution.

Table 3. Network VLANs

Network type	VLAN ID
PowerFlex Management	1109
PowerFlex Data 1	20
PowerFlex Data 2	30

Figure 4 shows a VMware network topology for the Tanzu implementation using the components in this solution: NSX Advanced Load Balancer and vSphere Networking, or more specifically the use of a vSphere Distributed Switch (VDS). The bottom half of the image references the Supervisor Cluster upon which the Tanzu Kubernetes clusters are built. The TKG boxes are the Kubernetes (Guest) cluster.

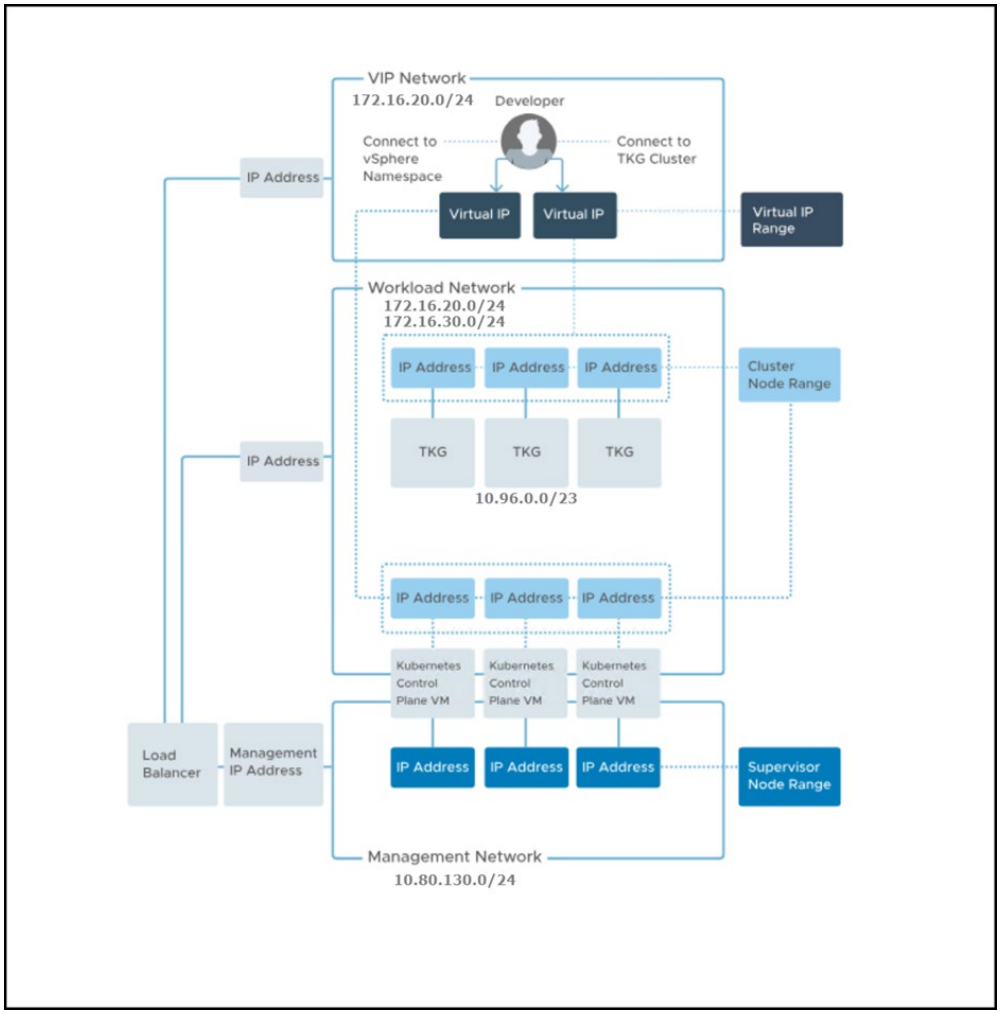

Figure 4. VMware-provided network topology for Tanzu with NSX Advanced Load Balancer

PowerFlex topology

The PowerFlex system consists of five nodes with the characteristics that are covered in Table 4. There is no SDC installed on any of the PowerFlex nodes as the volumes are only mapped to the ESXi hosts.

Table 4. PowerFlex node configurations

Model	MDM Role	SDS	SDT	CPU	Memory	Storage	Network
R650	Primary	Yes	Yes	2 * Intel(R) Xeon(R) Gold 6336Y CPU @ 2.40GHz (Core Count 24)	512 GB (16 * 32 GB, 2933 MT/s DRAM DDR-4), 32 GB (2 * 16 GB, 2933 MT/s NVDIMM-N DDR-4)	2 * 447.13 GB (SATA SSD), 10 * 1490.42 GB (SAS SSD)	2 * Mellanox ConnectX-5 EN 25GbE SFP28 Adapter (2 ports connected 10 Gb)
R650	Secondary	Yes	Yes	2 * Intel(R) Xeon(R) Gold 6336Y CPU @ 2.40GHz (Core Count 24)	512 GB (16 * 32 GB, 2933 MT/s DRAM DDR-4), 32 GB (2 * 16 GB, 2933 MT/s NVDIMM-N DDR-4)	2 * 447.13 GB (SATA SSD), 10 * 1490.42 GB (SAS SSD)	2 * Mellanox ConnectX-5 EN 25GbE SFP28 Adapter (2 ports connected 10 Gb)
R650	Tie Breaker	Yes	Yes	2 * Intel(R) Xeon(R) Gold 6336Y CPU @ 2.40GHz (Core Count 24)	512 GB (16 * 32 GB, 2933 MT/s DRAM DDR-4), 32 GB (2 * 16 GB, 2933 MT/s NVDIMM-N DDR-4)	2 * 447.13 GB (SATA SSD), 10 * 1490.42 GB (SAS SSD)	2 * Mellanox ConnectX-5 EN 25GbE SFP28 Adapter (2 ports connected 10 Gb)
R650	Secondary	Yes	Yes	2 * Intel(R) Xeon(R) Gold 6336Y CPU @ 2.40GHz (Core Count 24)	512 GB (16 * 32 GB, 2933 MT/s DRAM DDR-4), 32 GB (2 * 16 GB, 2933 MT/s NVDIMM-N DDR-4)	2 * 447.13 GB (SATA SSD), 10 * 1490.42 GB (SAS SSD)	2 * Mellanox ConnectX-5 EN 25GbE SFP28 Adapter (2 ports connected 10 Gb)
R650	Tie Breaker	Yes	Yes	2 * Intel(R) Xeon(R) Gold 6336Y CPU @ 2.40GHz (Core Count 24)	512 GB (16 * 32 GB, 2933 MT/s DRAM DDR-4), 32 GB (2 * 16 GB, 2933 MT/s NVDIMM-N DDR-4)	2 * 447.13 GB (SATA SSD), 10 * 1490.42 GB (SAS SSD)	2 * Mellanox ConnectX-5 EN 25GbE SFP28 Adapter (2 ports connected 10 Gb)

VMware topology

The VMware environment is a single vCenter with two clusters, Management and Workload, each with three ESXi hosts with the same configuration as the PowerFlex nodes. A visual depiction is in Figure 5.

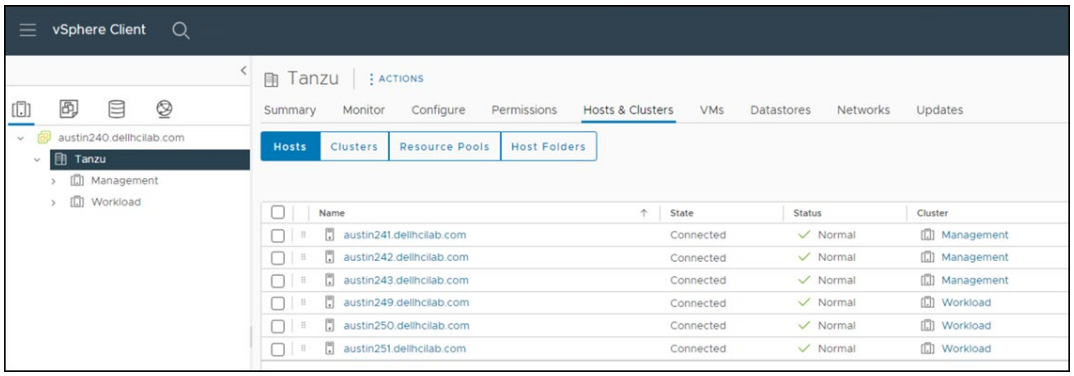

Figure 5. VMware vCenter

Protocol topology

Figure 6 and Figure 7 are logical depictions of the host connectivity between the VMware infrastructure and the PowerFlex. This solution uses both NVMe/TCP and SCSI (SDC) protocols to provision storage to the VMware environment. This concept holds true for both the Tanzu installation and the persistent storage that is created for Tanzu Kubernetes Clusters.

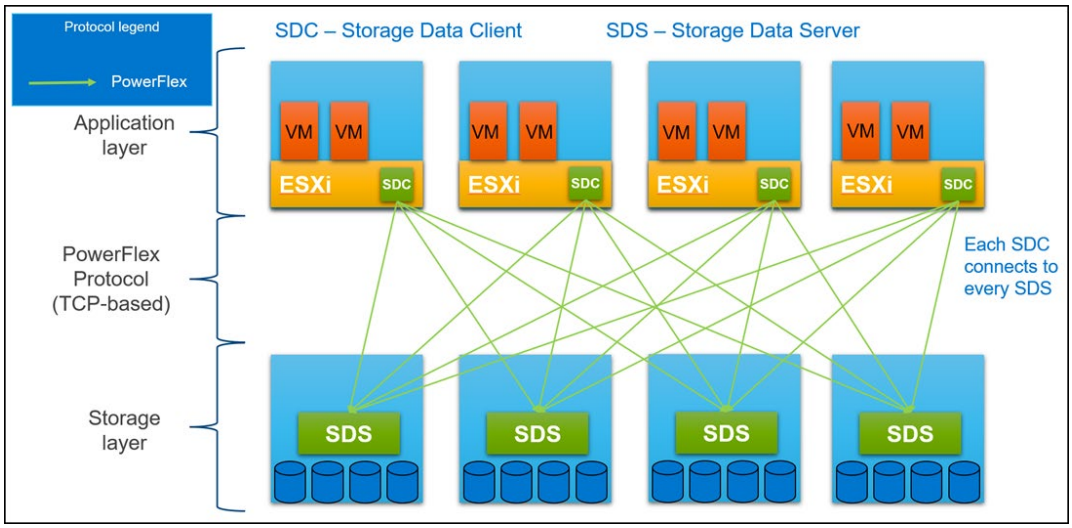

Figure 6. PowerFlex protocol host connectivity

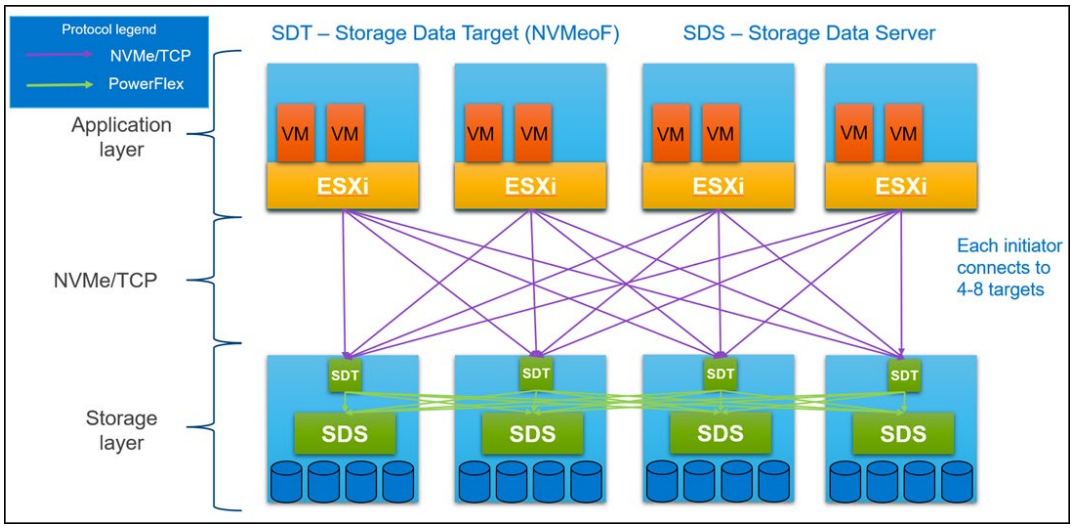

Figure 7. NVMe/TPC protocol host connectivity

Chapter 4 Host connectivity

This chapter presents the following topics:

Introduction

The following sections detail the setup of both SCSI (SDC) and NVMe/TCP as they are essential to this Tanzu solution.

Adding SDC to VMware

PowerFlex delivers the SDC as a vSphere Installation Bundle (VIB) that must be installed on ESXi. Although it is possible to do so manually, Dell offers the Virtual Storage Integrator (VSI) which automates the process.

VSI

The following procedures use VSI to add the storage systems.

Add the PowerFlex storage systems

1. Register the PowerFlex environment in VSI. In vCenter, go to **Dell VSI** in the main menu as shown in Figure 8.

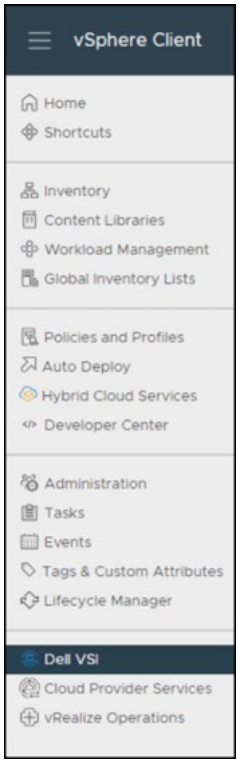

Figure 8. Dell VSI

2. Select the **Storage Systems** option as shown in Figure 9, and then click the **plus (+)** button.

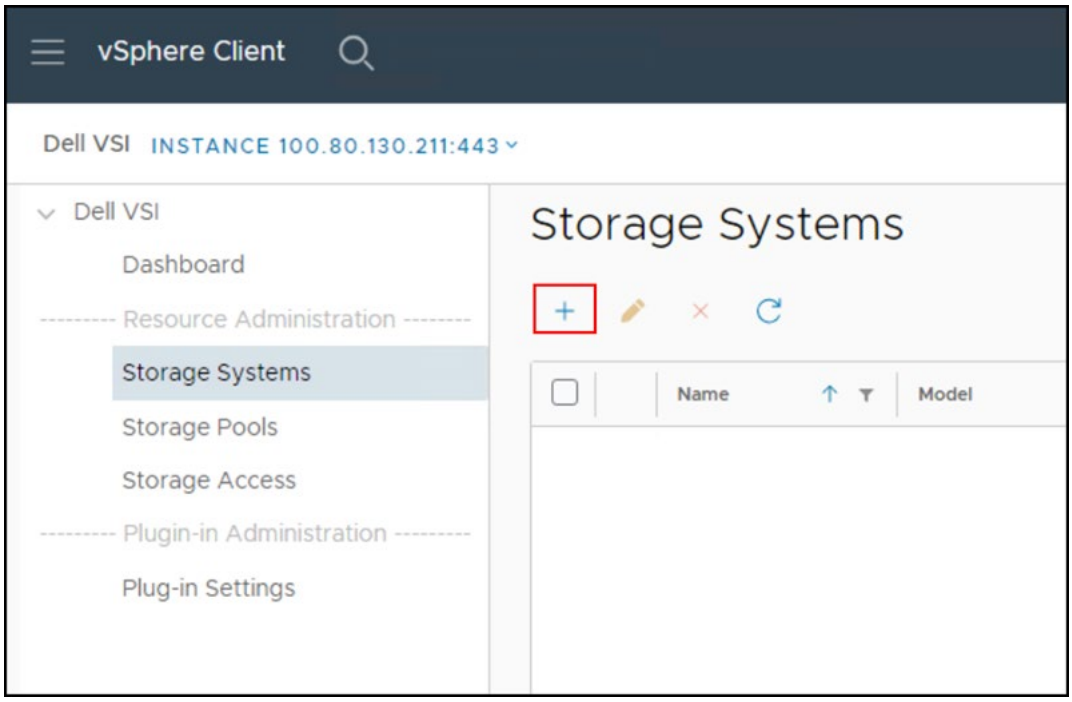

Figure 9. Dell VSI: Add PowerFlex storage system

3. Choose **PowerFlex** from the drop-down storage systems as shown in Figure 10.
4. Enter the appropriate information.
5. Select **NEXT**.

Note: The **PowerFlex Gateway** for 4.0.x version is the same as the UI IP.

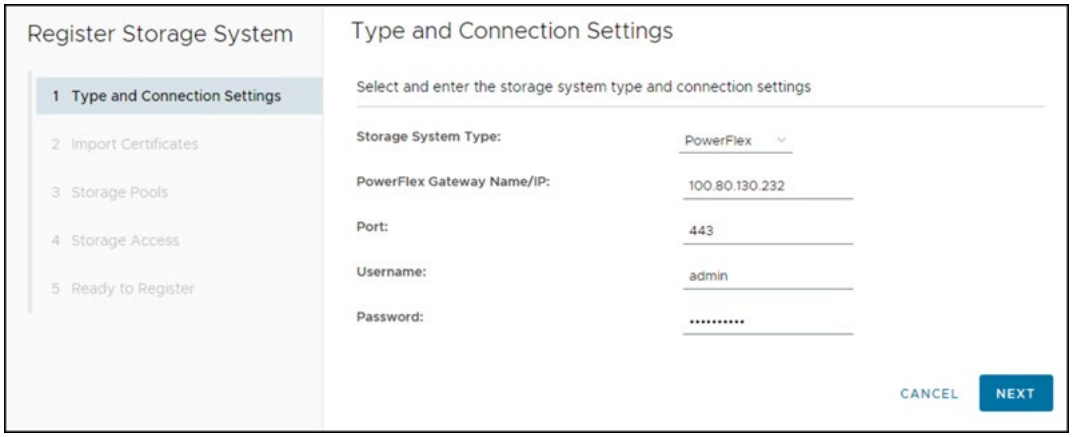

Figure 10. DELL VSI: Add PowerFlex storage system – Step 1

6. Choose whether to import certificates.
7. Select **NEXT**.
8. VSI then displays all available storage pools on the PowerFlex that are available to select.

Note: Some customers may have separate pools for Tanzu work. In this example, there is only one pool.

9. Select **NEXT**.
10. Choose whether any user other than an administrator can use some VSI functionality.
11. Select **NEXT**.
12. Review your selections.
13. Select **FINISH**.
14. The system is now available in the **Storage Systems**. Select it to view the details as shown in Figure 11.

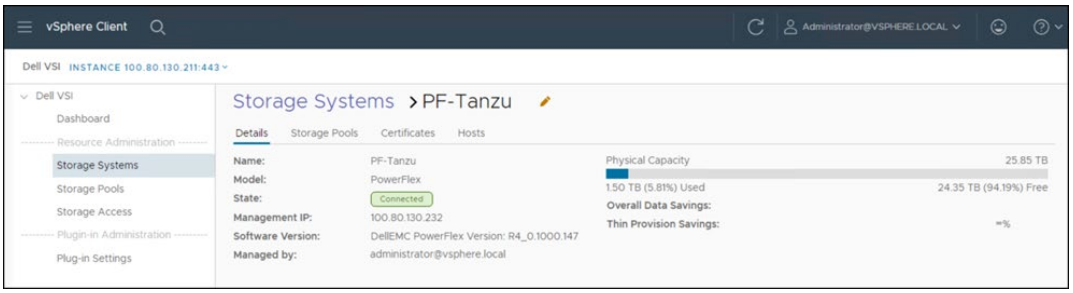

Figure 11. Dell VSI: Add PowerFlex storage system – PF-Tanzu

Add the SDC

1. After the PowerFlex is added to VSI, select the **Hosts** tab. Here, in Figure 12, VSI recognizes that three hosts are available for an SDC installation.
2. Click the **plus sign** to begin the SDC installation process.

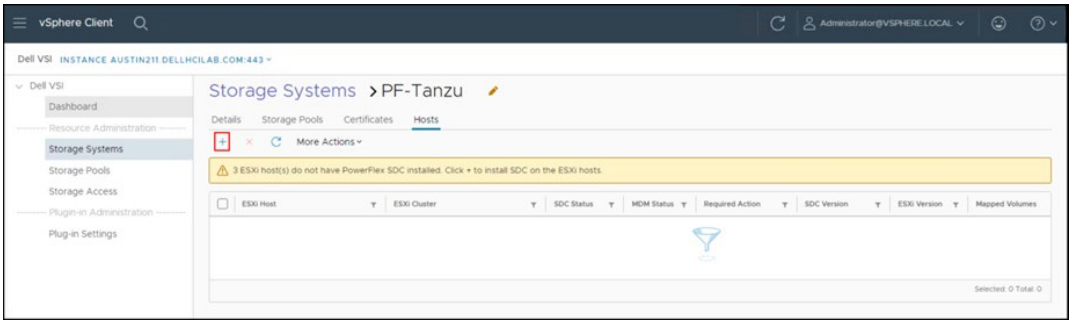

Figure 12. Dell VSI: Add SDC – Step 1

3. To install the SDC using VSI, place the .zip file (for example, `sdc-4.0.0.1211-esx7.x.zip`) in the same location on every ESXi host.

Note: This procedure can be done individually on each host, or by using a shared datastore. The environment may not include other arrays to create a shared VMFS. In these cases, Dell Technologies advises creating a Windows share which can then be mounted as an NFS datastore on each ESXi host. Doing so avoids keeping a copy of the .zip file on each host and guarantees that every host uses the same file.

4. Click **INSTALL**.

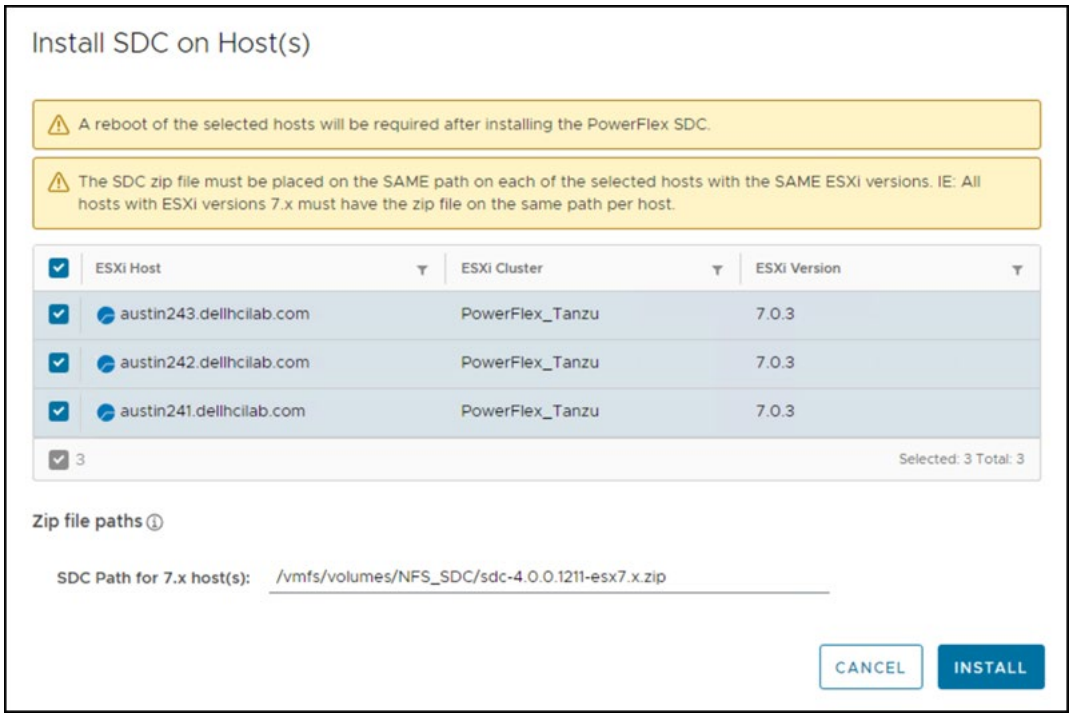

Figure 13. Dell VSI: Add SDC – Step 2

5. When the installation is complete, VSI displays a message, as shown in Figure 14, stating that the hosts must be rebooted.

Note: VSI **does not** automatically reboot the hosts.

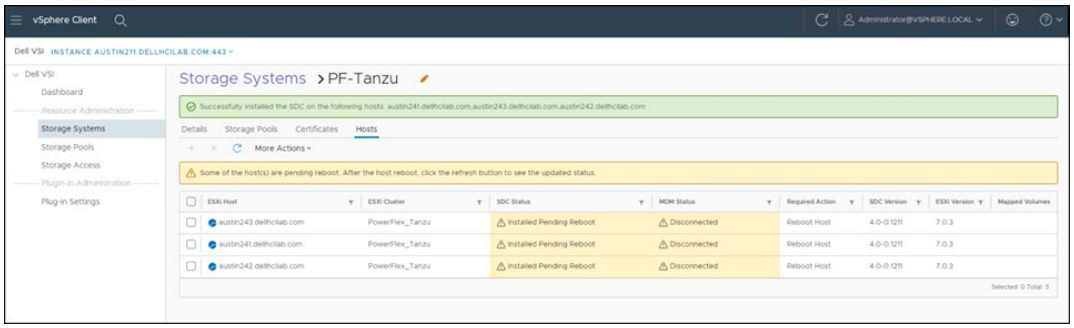

Figure 14. Dell VSI: Add SDC – Step 3

6. Reboot the hosts.
7. After the host comes back up, return to the **Hosts** screen to see the updated status, as shown in Figure 15. VSI updates the parameters on the ESXi host in the **scini** module which connect the host to the registered PowerFlex (MDM) environment.

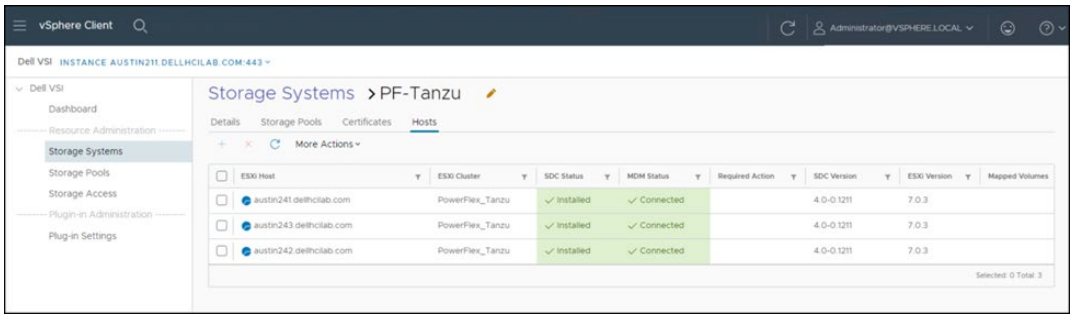

Figure 15. Dell VSI: Add SDC – Complete

Rename the hosts

When the SDC install completes, Dell Technologies recommends renaming the host in the PowerFlex UI for ease of use. Doing so is important if using both SDC and TCP on the same host.

Note: VSI is only a green-field option since it cannot resolve issues that are related to improper installations. If any of the ESXi hosts in the cluster have unresolved SDC components on them already, VSI displays the following error.

Figure 16. Error constructing hostname

Adding NVME/TCP to VMware

Introduction

While VSI can install SDC for users, it cannot configure NVMe/TCP on the ESXi host. It is a manual process that users must complete on each ESXi host that require access to storage through this protocol. The following procedures walk through this configuration on one of the ESXi hosts in the Tanzu cluster.

Add the NVME/TCP adapter

As covered in Protocol topology, NVMe/TCP does not require a separate installation on the ESXi host. Rather, the NVMe/TCP software adapter attaches to the NVMe target (SDT process) on one or more PowerFlex nodes. In the PowerFlex system used in this guide, the SDT process uses the same Ips as SDC. Therefore, the data networks of VLAN 50 and 100 are used.

1. Before adding a NVMe/TCP adapter, determine the physical adapters for each VLAN.
2. Select the **NVMe over TCP service** on the VMkernel as shown in Figure 17.
3. Select **OK**.

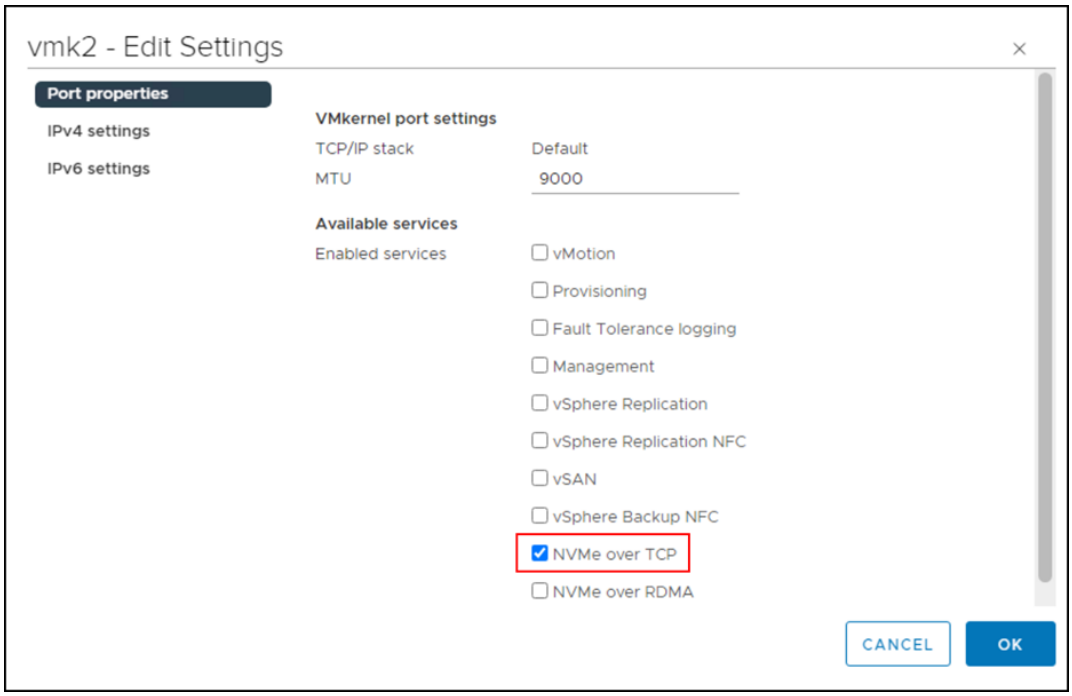

Figure 17. Enable the NVMe/TCP service on the VMkernel

4. With the service enabled, go to **Host -> Configure -> Storage Adapters -> ADD SOFTWARE ADAPTER**.
5. Use the drop-down to pick the correct physical NIC for the selected VLAN.

Add the NVMe host to PowerFlex

Before adding the controllers to the adapter, you must create the NVMe host on the PowerFlex by obtaining the host NQN from vSphere.

1. Go to **Host -> Configure -> Storage Adapters**.
2. Select the newly added storage adapter as shown in Figure 18.

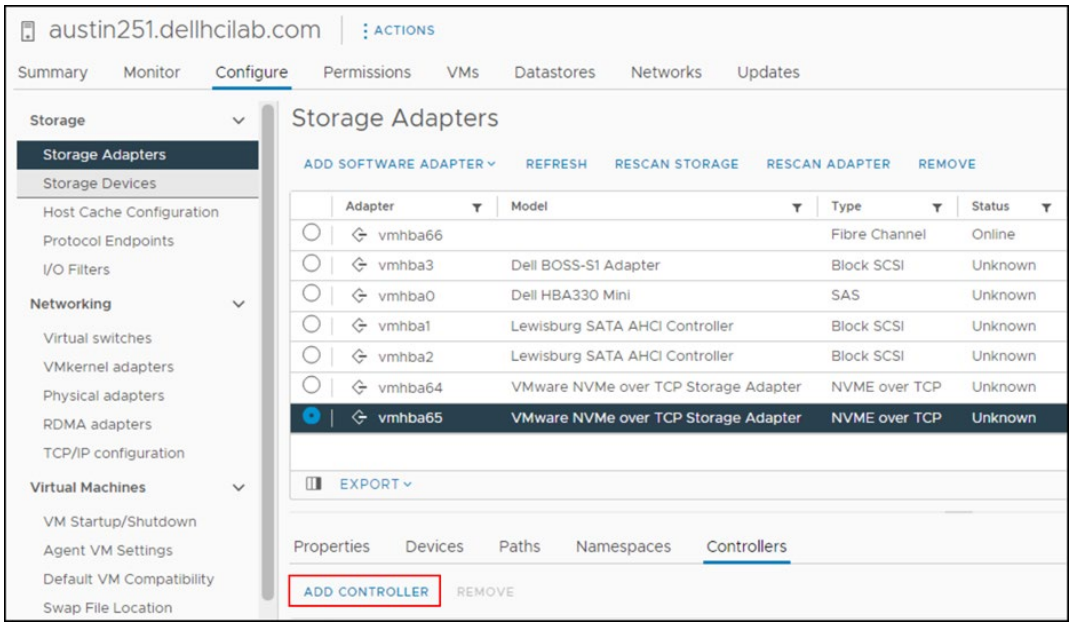

Figure 18. Select the storage adapter

3. Select the **Controllers** tab.
4. Select **ADD CONTROLLER**.
5. In the screen that follows, select the **COPY** button next to the **Host NQN** as shown in Figure 19.

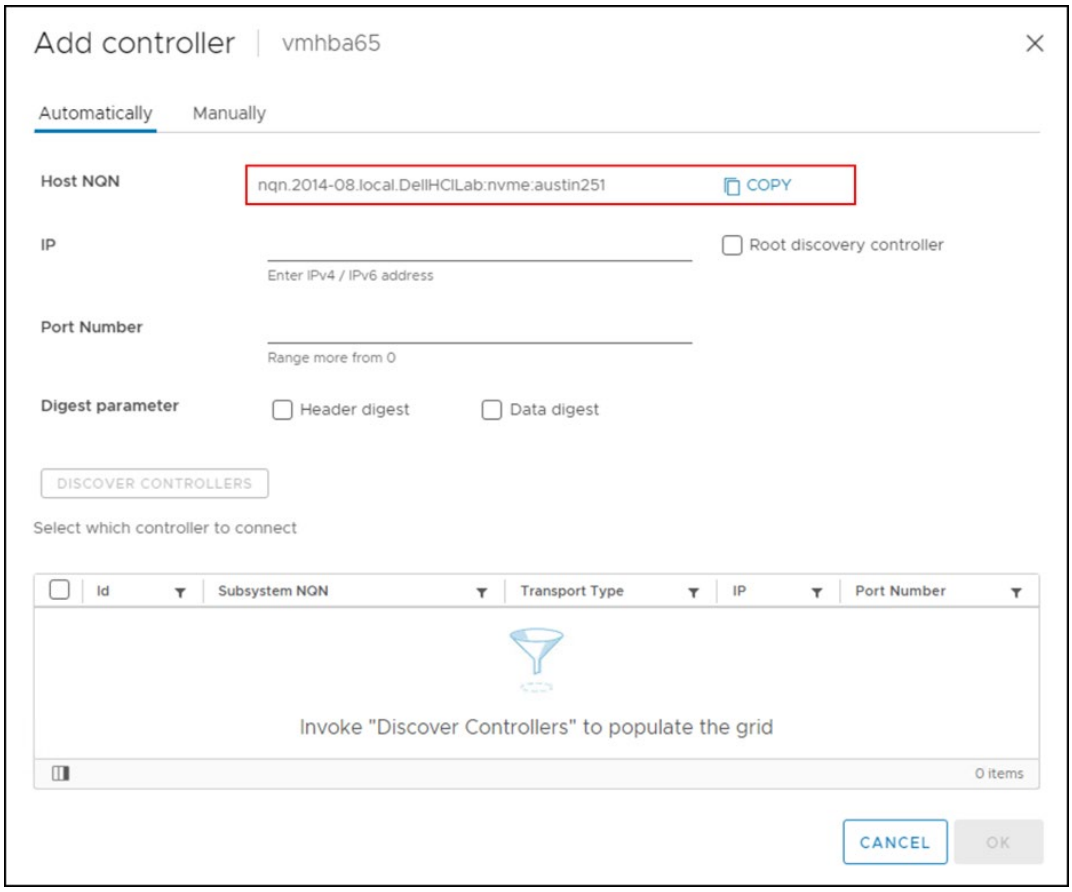

Figure 19. Copy host NQN

6. Open the PowerFlex management UI and go to the **Block -> Hosts** menu.
7. Select **+ Add Host**.
8. Paste the copied host NQN into the **Host NQN** field and provide a **Host Name**.
9. Leave the other fields as default, as shown in Figure 20.

Note: Since both SDC and NVMe/TCP are being used, the suffix "*_nvme*" is added to the hostname.

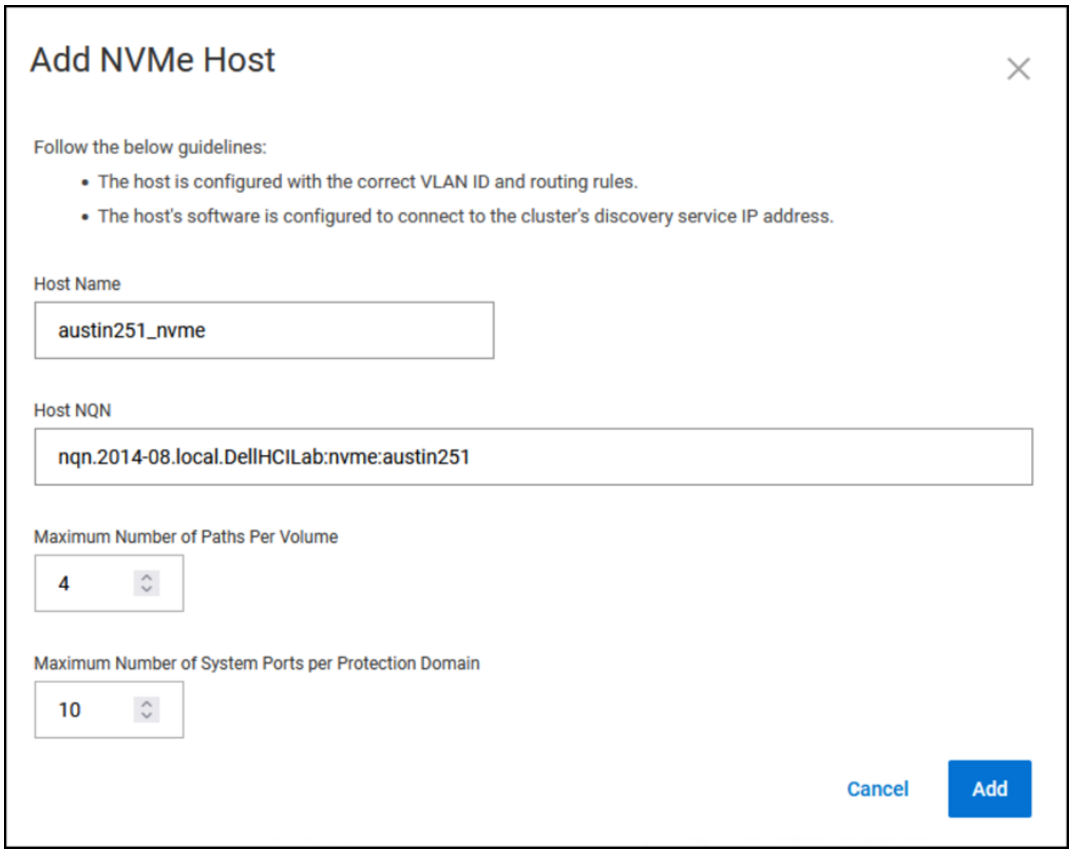

Figure 20. Add NVMe host

Add the NVMe controller

The SDT IP addresses are required in order to add the controller. These addresses can be obtained from the storage admin or by users with access to the PowerFlex UI.

1. While still in the PowerFlex UI, go to **Block -> NVMe Targets**.
2. Select one of the SDTs listed, and on the right side find the IP addresses. These addresses are the controller IP addresses
3. Return to vSphere.
4. Go to the **ADD CONTROLLER** screen as shown in Figure 19.
5. Enter the copied IP address into the **IP** field.
6. Enter 8009 into the **Port Number** field (4420 also works).
7. Select **DISCOVER CONTROLLERS**. The NVMe target is displayed.
8. Select the **NVMe target**, and then click **OK**, as shown in Figure 21.

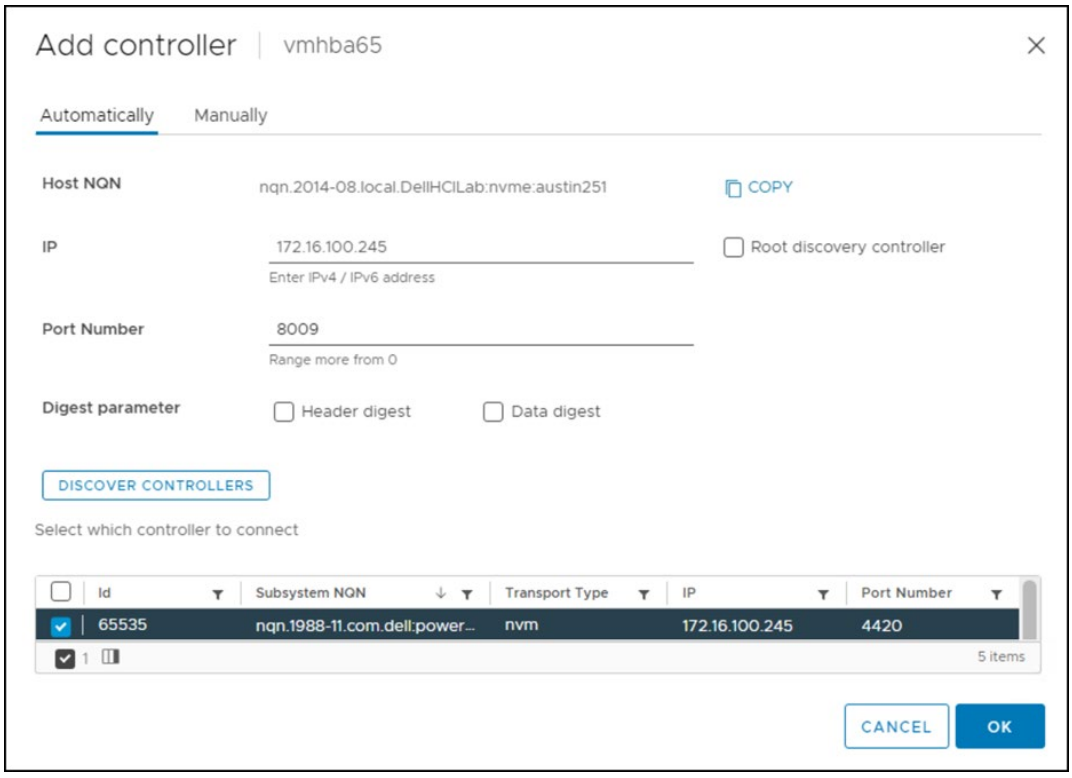

Figure 21. Add controller

9. Repeat this process for each of the NVMe target Ips in VLAN 100.
10. Add a second NVMe/TCP adapter for VLAN 50.
11. You can now map devices to the ESXi host.

Provision storage

Introduction

With connectivity to the ESXi hosts established, you can create and map volumes for the VMware datastores. For both SDC and NVMe/TCP this task can be accomplished directly in the PowerFlex UI. However, VSI enables users to provision datastores from the vCenter for SDC mappings. VSI does not support NVMe/TCP (or NFS). An example of each follows, beginning with the PowerFlex UI.

NVMe/TCP datastores

1. Log in to the PowerFlex UI.
2. Go to **Block -> Volumes**.
3. Select **+ Create Volume** as shown in Figure 22.

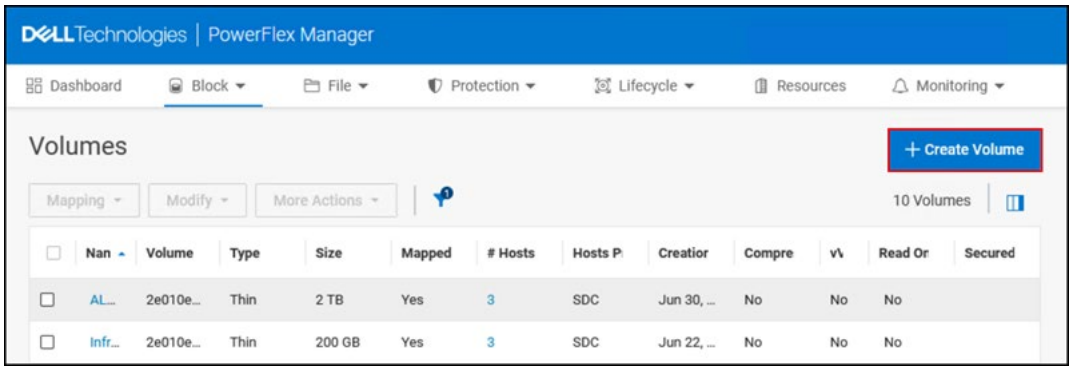

Figure 22. NVMe/TCP datastores – Step 2

4. In the Create Volume screen, the **Number of volumes** defaults to **1**.
5. Enter a **Volume name**.
6. Leave **Provisioning** as the default **Thin**.
7. Select a **size**. This example uses 2 TB.
8. Select the **Storage Pool**.
9. Click **Create** to complete the wizard.
10. After volume creation, use the **check box** to select the new volume.
11. From the **Mapping** menu, select **Map** as shown in Figure 23.

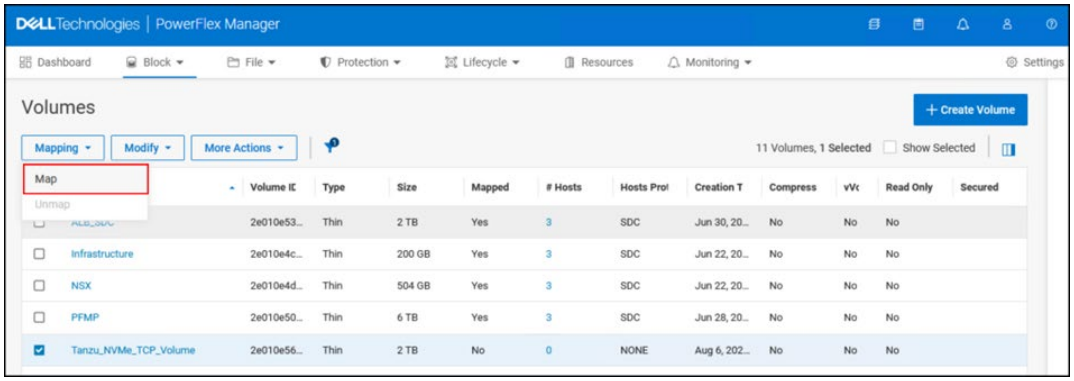

Figure 23. NVMe/TCP datastores – Step 4

12. Use the radio button at the top to choose the **NVMe** protocol.
13. Using the check boxes, select the **NVMe hosts** where to map the volume.
14. Click **Map**.

Note: The volume will be available to the hosts for datastore creation after mapping.

15. Go to any of the ESXi hosts.
16. Select **Configure -> Storage -> Storage Adapters**.
17. Select one of the **NVMe/TCP adapters** to view the newly presented device.
18. Log in to vCenter.

19. From the data center level, go to **Datastores**, then go to **ACTIONS -> Storage -> New Datastore**.
20. Use the **radio button** to select the newly presented TCP volume.
21. Provide a **Name**.
22. Select **NEXT** to complete the wizard.

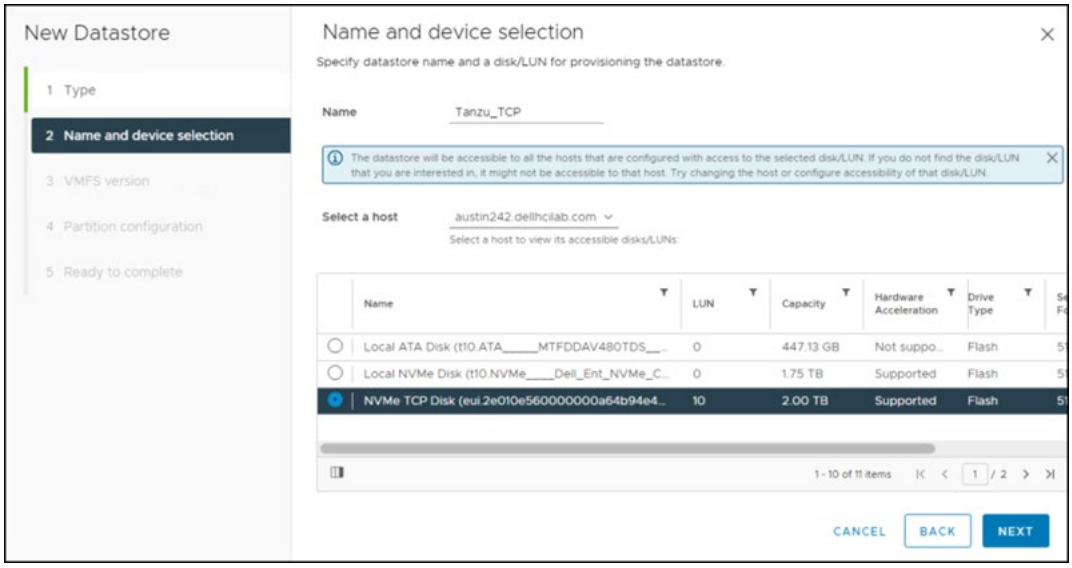

Figure 24. NVMe/TCP datastores – Step 8

The remaining steps are the same for TCP as any other device. The result is shown in Figure 25.

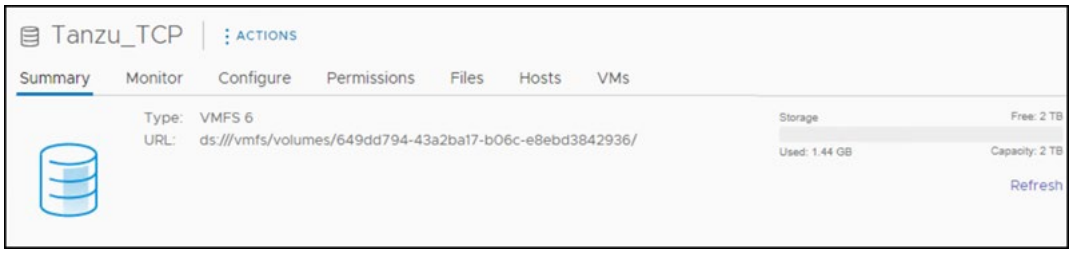

Figure 25. NVMe/TCP datastores – Complete

SDC datastores

For datastores with backing volumes that are mapped to SDC:

1. In the vSphere UI, right-click at the data center, cluster, or host level.
2. Select **Dell VSI -> Create Datastore**. This example is shown at the data center level in Figure 26.

Note: This task starts the **Create Datastore** wizard, which is a combination of VSI and VMware functionalities.

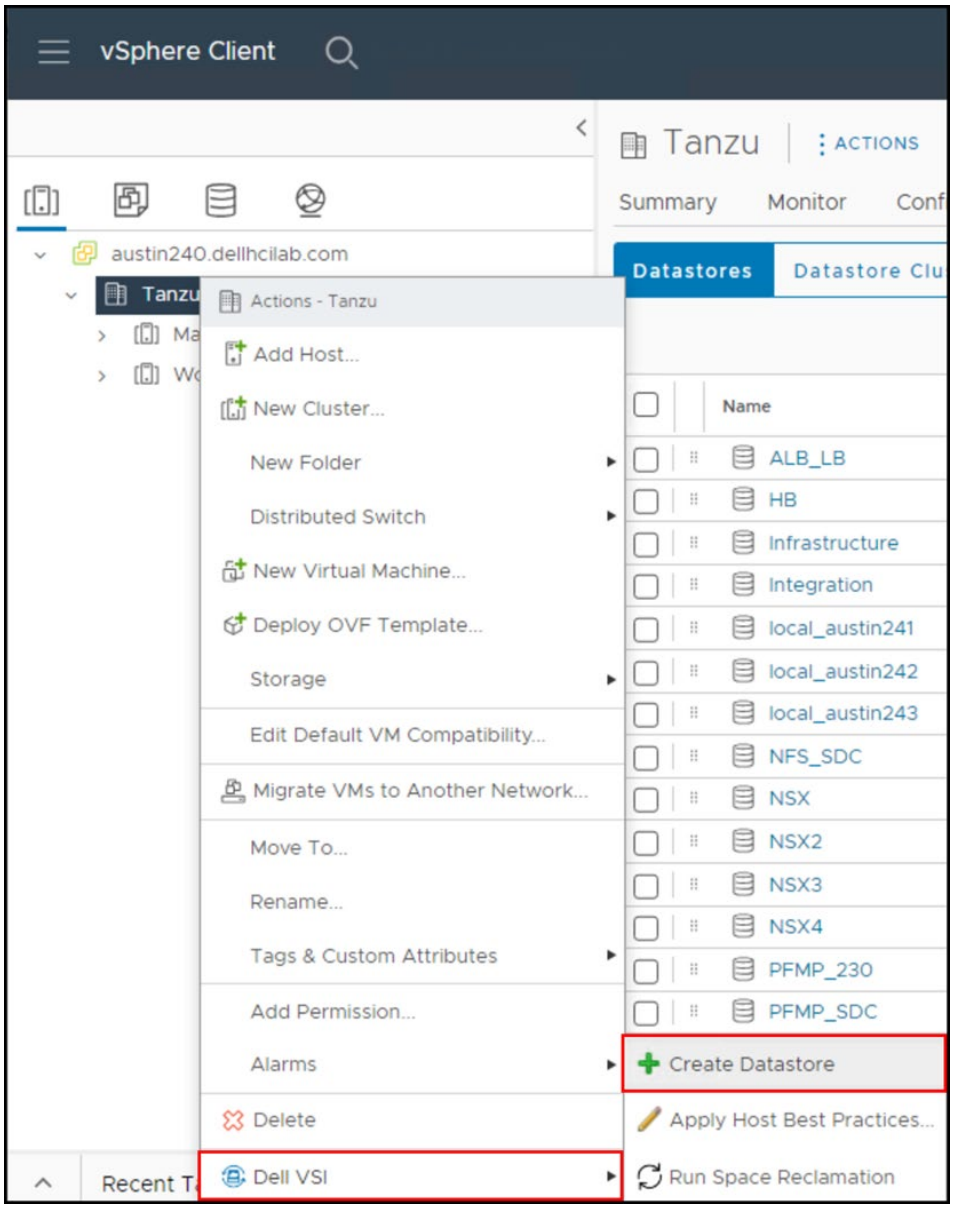

Figure 26. SDC datastores – Start VSI wizard

3. Select **VMFS** for the type of datastore to create, as shown in Figure 27.

Note: VMFS is the only type that is supported for PowerFlex on VSI as of the publication of this guide.

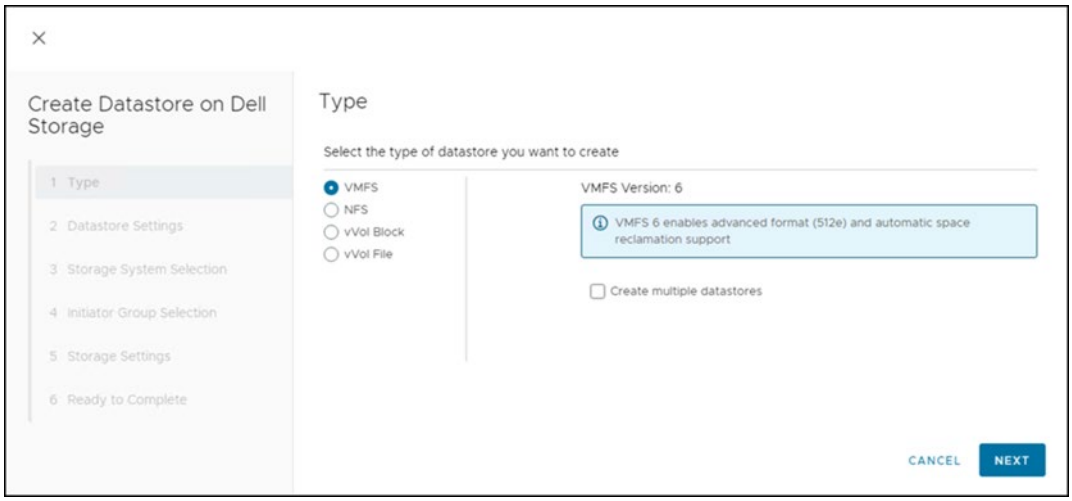

Figure 27. SDC datastores – Step 1

4. Click **NEXT** to continue.
5. Enter a **Name** for the datastore.
6. Click **NEXT**.

VSI displays the available storage systems that the user registered in VSI. These systems could include PowerFlex and non-PowerFlex storage. In this example, there is a single PowerFlex storage system, named **PF-Tanzu**.

7. Select a storage system as shown in Figure 28.

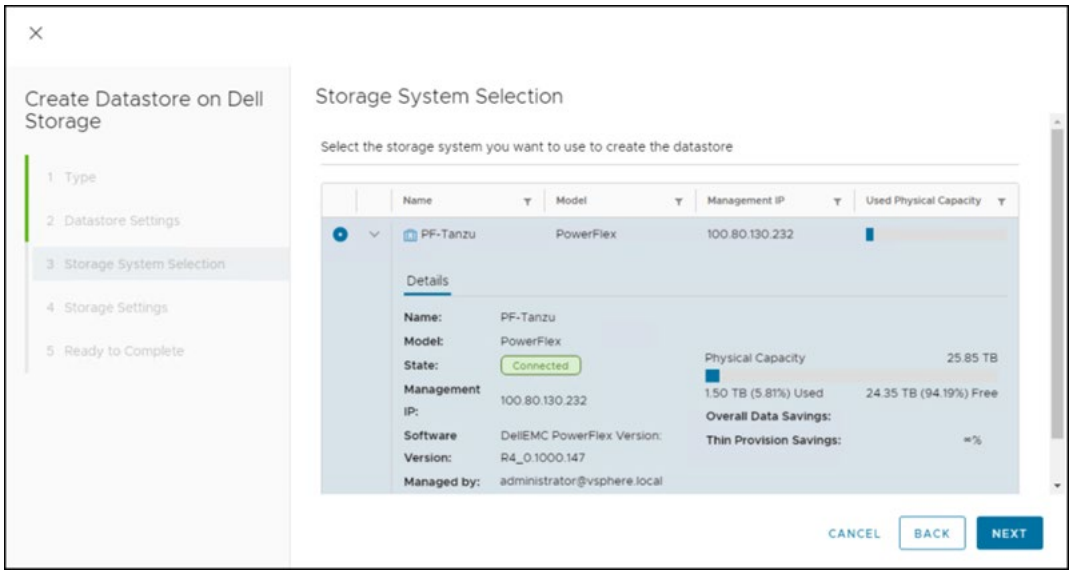

Figure 28. SDC datastores – Step 3

8. Click **NEXT**.

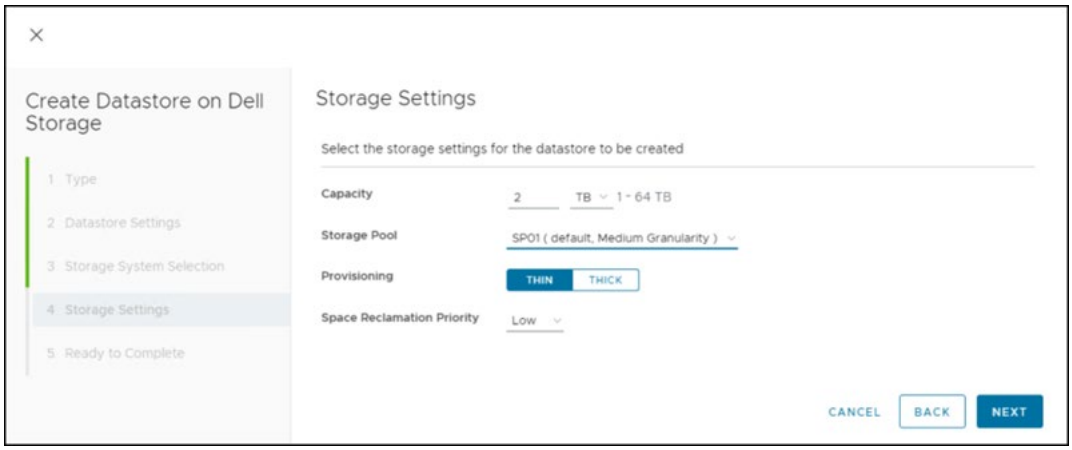

Figure 29. SDC datastores – Step 4

9. Under **Storage Settings**, enter a capacity for the datastore.
10. Use the drop-down box to pick the **storage pool**.

Note: Only one pool is available in this example.

11. Click **NEXT.**
12. Review the input information.
13. Click **FINISH**. VSI then performs the following:
    a. Provisions the volume
    b. Maps it to all SDC hosts that are part of either the data center, cluster, or ESXi host, depending on what was selected at the start
    c. Creates a datastore on it
14. Find the newly created datastore.
15. Go to **Configure -> Dell VSI -> Configure Storage**.

The **Storage Settings** tab defaults with information about the volume, as shown in Figure 30.

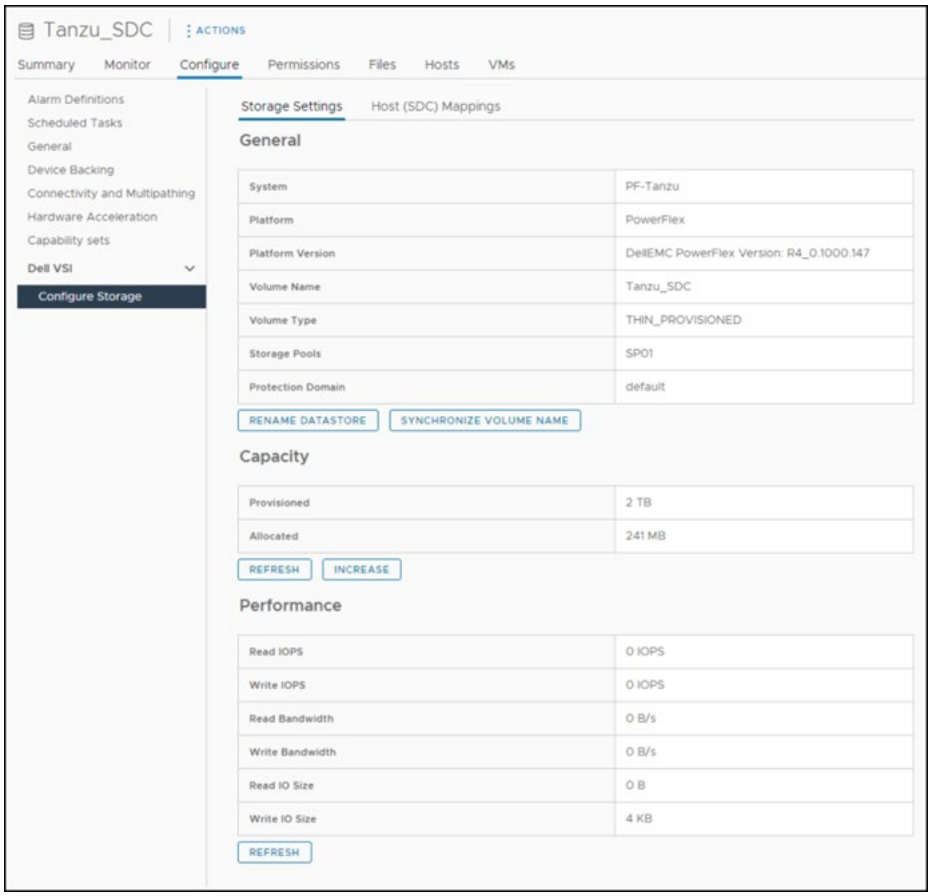

Figure 30. Tanzu_SDC datastore in VSI – Storage Settings

16. To see the SDC hosts to which VSI mapped the volume, select the **Host (SDC) Mappings** tab as shown in Figure 31.

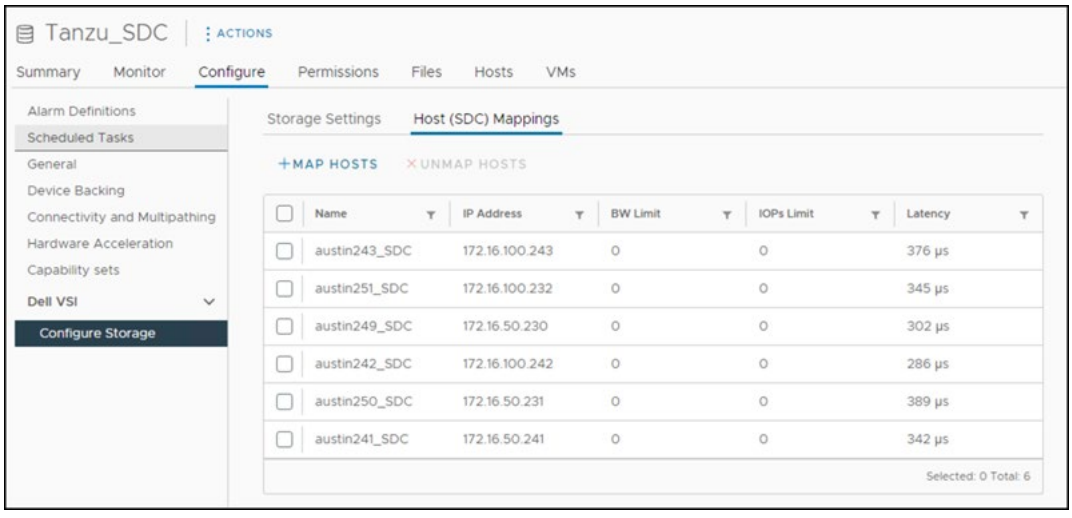

Figure 31. Tanzu_SDC datastore in VSI – Host (SDC) Mappings

Chapter 5 Configuring VMware infrastructure

This chapter presents the following topics:

Introduction

As part of the Tanzu setup, the user must setup storage policies within vCenter. A storage policy is an object that encompasses defined rules or qualities about storage. These rules dictate where to create a VM or even .vmdk (for example, first class disk). For example, a storage policy might be created that says the VM or .vmdk must be created on an SSD datastore. Tanzu uses these policies when creating persistent storage for the containers by converting them into storage classes.

In this Tanzu setup, tagging is used which enables the user to specify exactly which datastores should be in a policy. There is no ability to create a storage policy that is only associated with SDC-presented volumes or NVMe/TCP-presented volumes. Instead, create a “tag” and manually assign it to the datastores based on the protocol.

Note: The naming of storage policies in VMware is flexible, for example, upper case, spaces, and so on. Storage class names that are associated with these policies are not. Therefore, VMware automatically converts noncompliant storage policy names when creating storage classes. For example, a storage policy that is named TCP policy is changed to the **tcp-policy** storage class. Users may be better served in adhering to the storage class naming restrictions when creating storage policies, unlike the guide.

VMware storage policies

Introduction

The following sections include the creation of two storage policies for the Tanzu environment. Both category and tag creation are covered as precursors to the policy creation. A category is required before the user creates tags because tags are assigned a category.

Note: A category can be shared among multiple tags.

Category

1. In vSphere, click the **hamburger menu** ( ) on the left.
2. Select **Tags & Custom Attributes**.
3. Select **NEW** under the **Tags-CATEGORIES** option as shown in Figure 32.

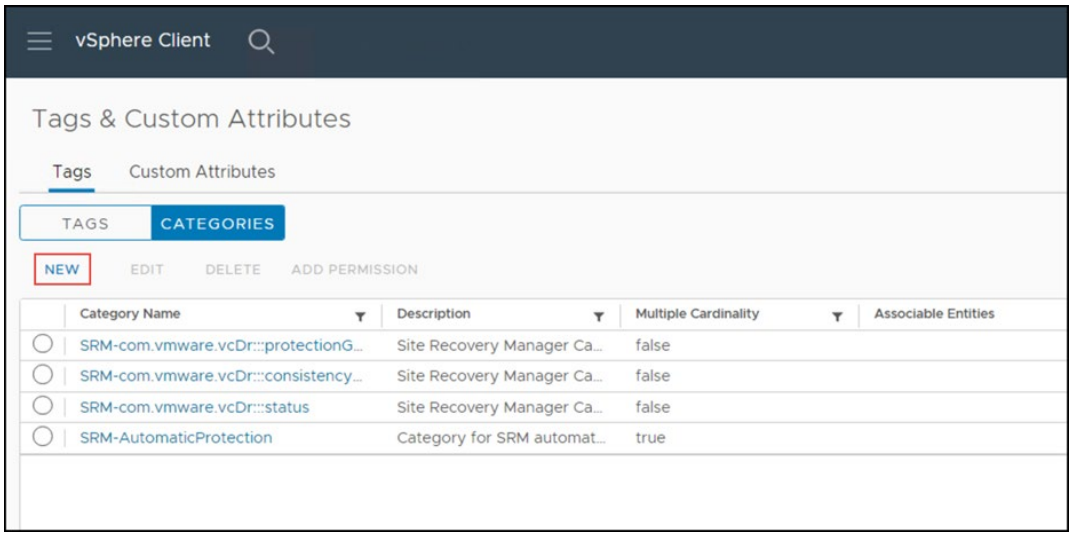

Figure 32. New category

4. Enter a **Name** for the category and choose whether to use one or many tags.
5. Optionally, add a **Description**.

Note: Figure 33 shows an example in which a single tag is used to avoid applying multiple options to the same datastore.

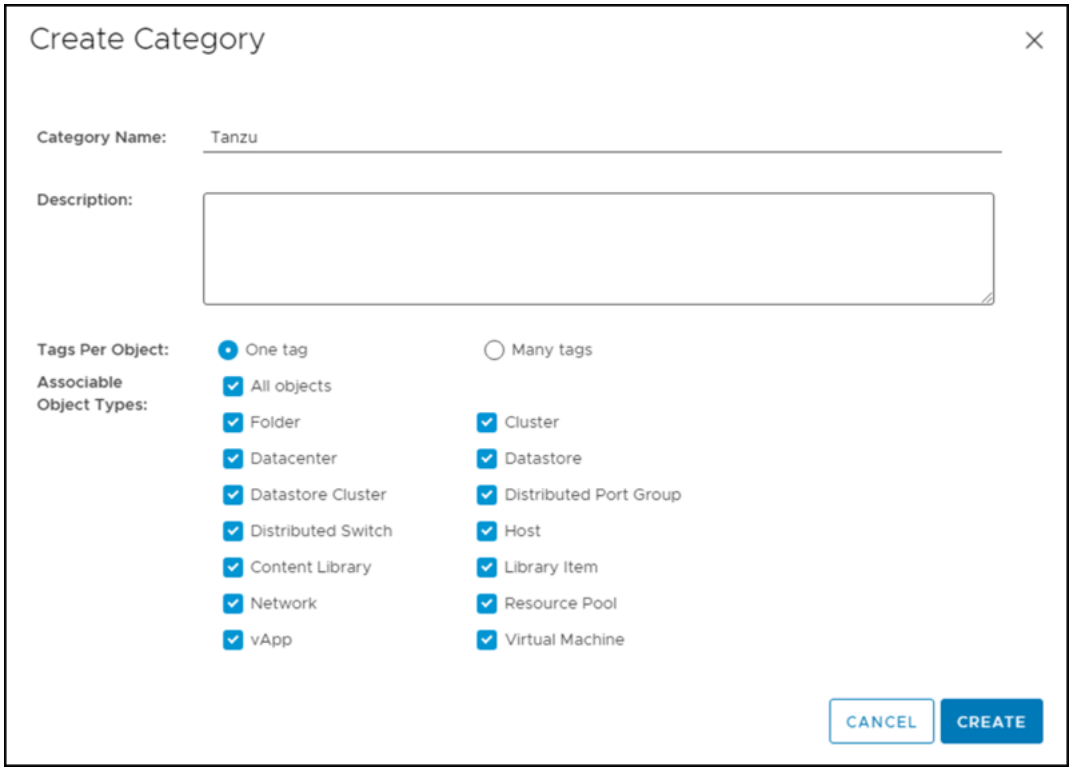

Figure 33. Create a category

Tags

Since both SDC and NVMe/TCP is used for PowerFlex volumes, two tags are necessary.

SDC

1. Return to the **Tags & Custom Attributes** screen.
2. Select the **TAGS** button, and then and select **NEW** to display the Create Tag dialog window.

Figure 34. Add an SDC tag

3. Enter a **Name** for the tag. This example uses **SDC**.
4. Use the drop-down box to choose the **Tanzu** category previously created.
5. Click **Create** to complete the tag.

NVME/TCP

1. Return to the **Tags & Custom Attributes** screen.
2. Select the **TAGS** button, and then select **NEW**.

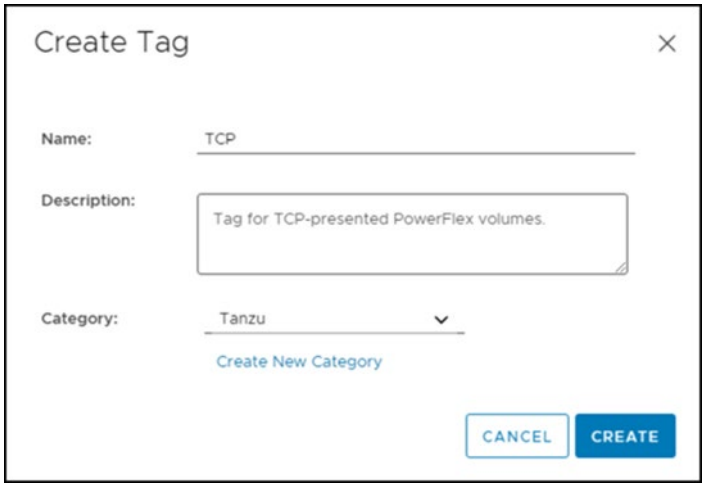

Figure 35. Add a NVMe/TCP tag

3. Enter a **Name** for the tag. This example uses **TCP**.
4. Use the drop-down box to choose the **Tanzu** category previously created.
5. Select **Create** to complete the tag.

Datastore assignment

You can now assign the tag to the appropriate datastores. This process is manual as explained earlier, because the tags do not inherit any attributes from the array. A tag is whatever you want it to be, and it can be assigned to any number of objects.

Start by identifying the two different types of datastores on the PowerFlex volumes.

SDC

1. In vSphere, go to the vCenter **Datastores -> Datastores** view.
2. Right-click the SDC datastore and go to the **Tags & Custom Attributes -> Assign Tag…** option, as shown in Figure 36.

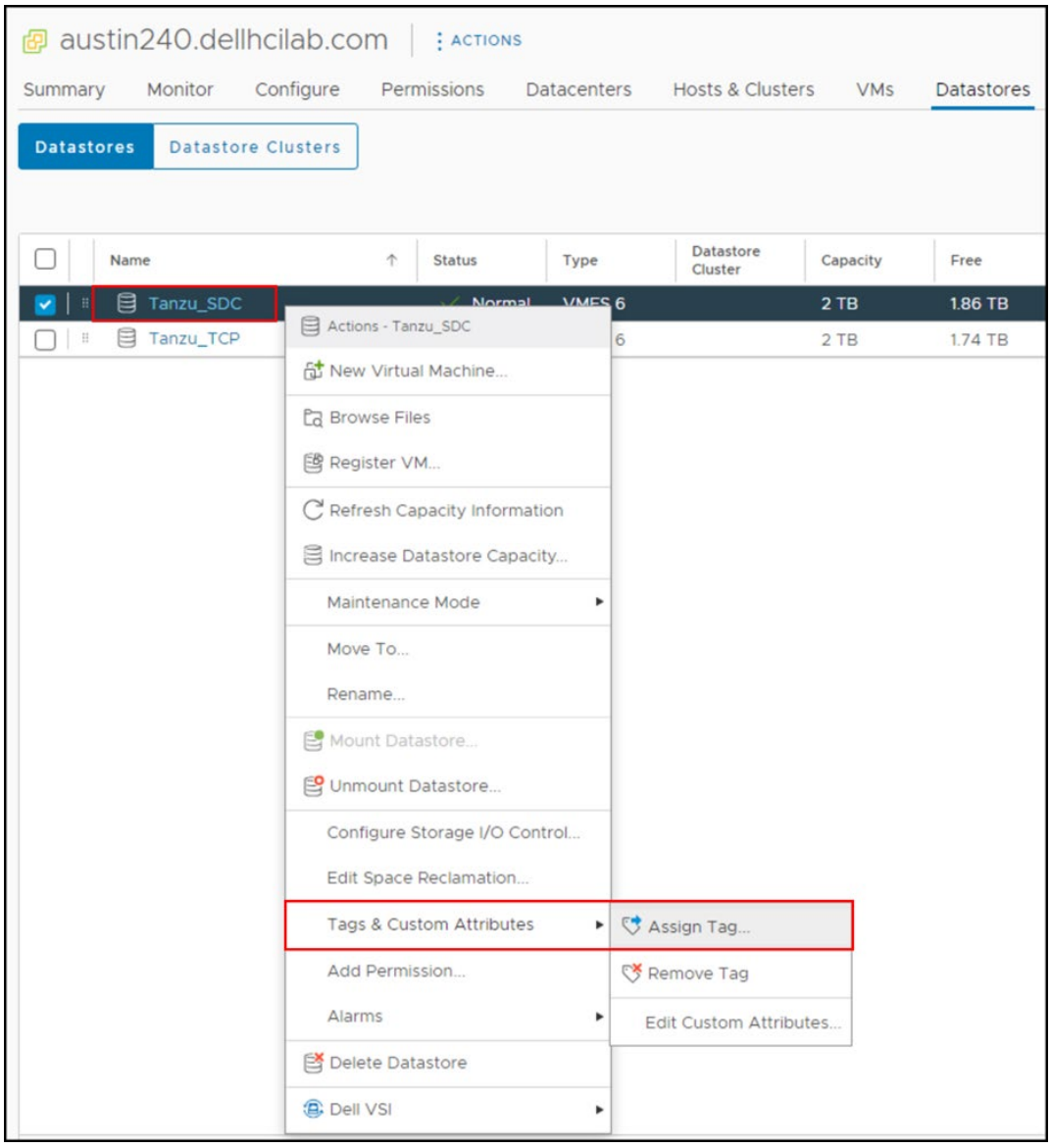

Figure 36. Assign SDC tag – Step 1

3. Use the check box to select the **SDC** tag as shown in Figure 37.

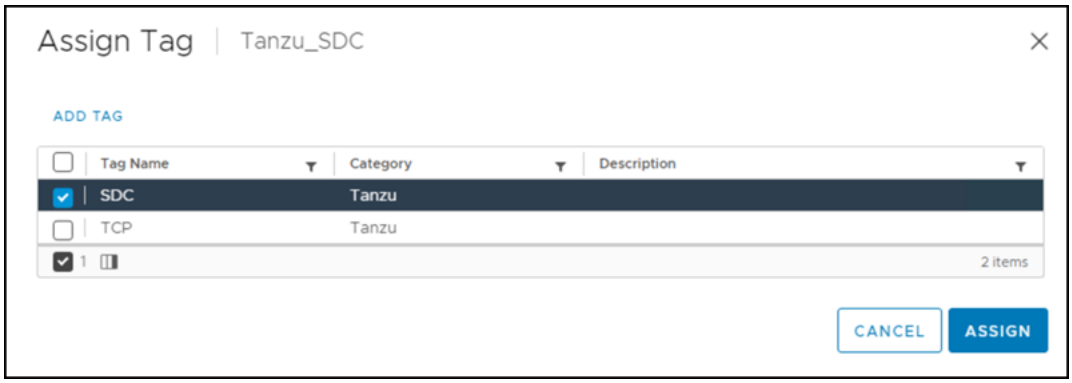

Figure 37. Assign SDC tag – Step 2

TCP

In vSphere, repeat the same process.

1. Right-click the TCP datastore and go to the **Tags & Custom Attributes -> Assign Tag…** option.
2. Use the check box to select the **TCP** tag.

Storage policies

The tag-based storage policy creation is the final step in the process. The SDC storage policy is shown as an example, but the TCP policy is included at the end of the process.

SDC

1. In vSphere, click the **hamburger menu** ( ) on the left.
2. Go to the **Policies and Profiles** menu.
3. Select **VM Storage Policies** as shown in Figure 38.

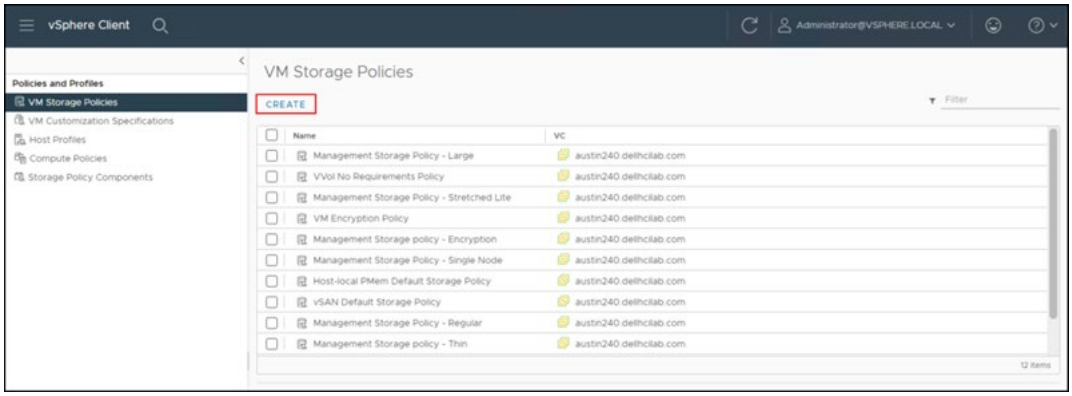

Figure 38. SDC storage policy – Step 1

4. Click **CREATE**.
5. Enter a **Name** (required) and **Description** (optional) for the policy.
6. Click **NEXT**.
7. Under the Policy structure screen, check the box next **to Enable tag based placement rules**, as shown in Figure 39.

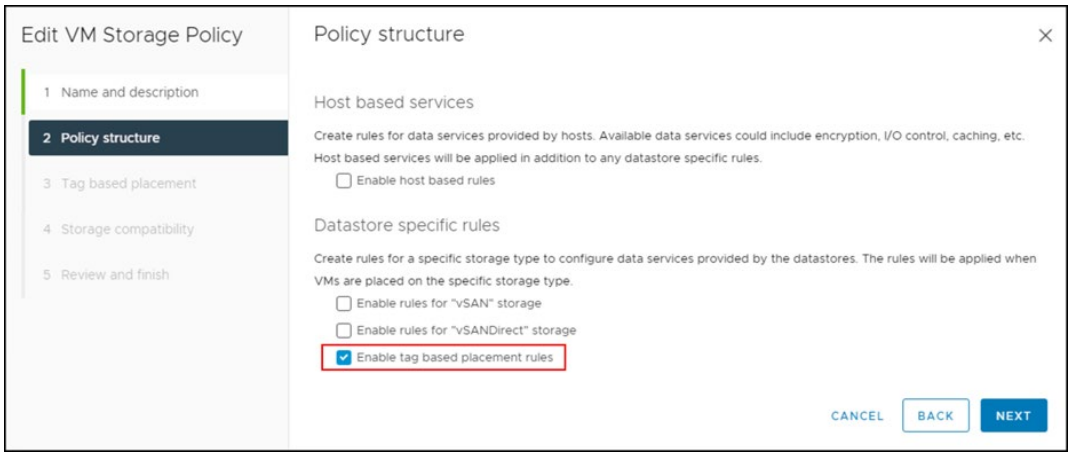

Figure 39. SDC storage policy – Step 3

8. Click **NEXT**.
9. Now that tagging is enabled, select the **category** that was created earlier from the drop-down box as shown in Figure 40.

Note: Doing so enables the **BROWSE TAGS** button.

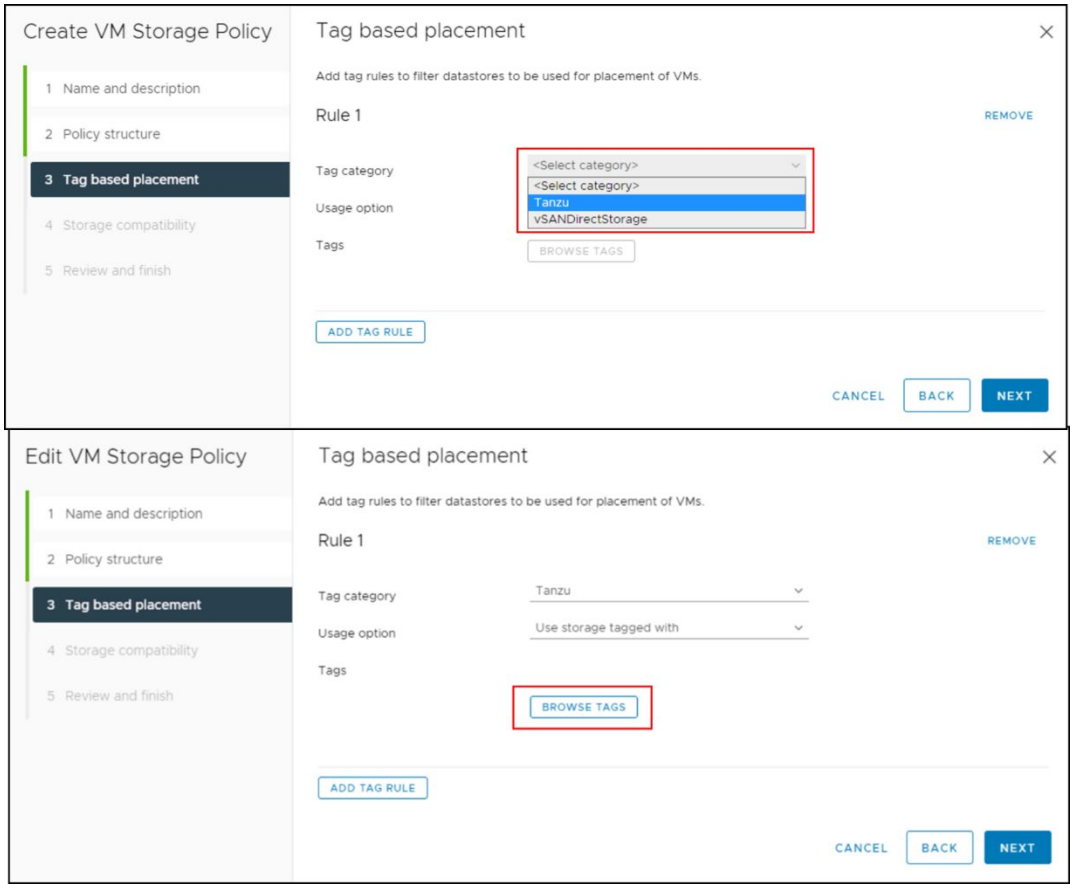

Figure 40. SDC storage policy – Step 4

10. Click **BROWSE TAGS**.

11. Use the **check box** next to the correct tag, and then click **OK** as shown in Figure 41.

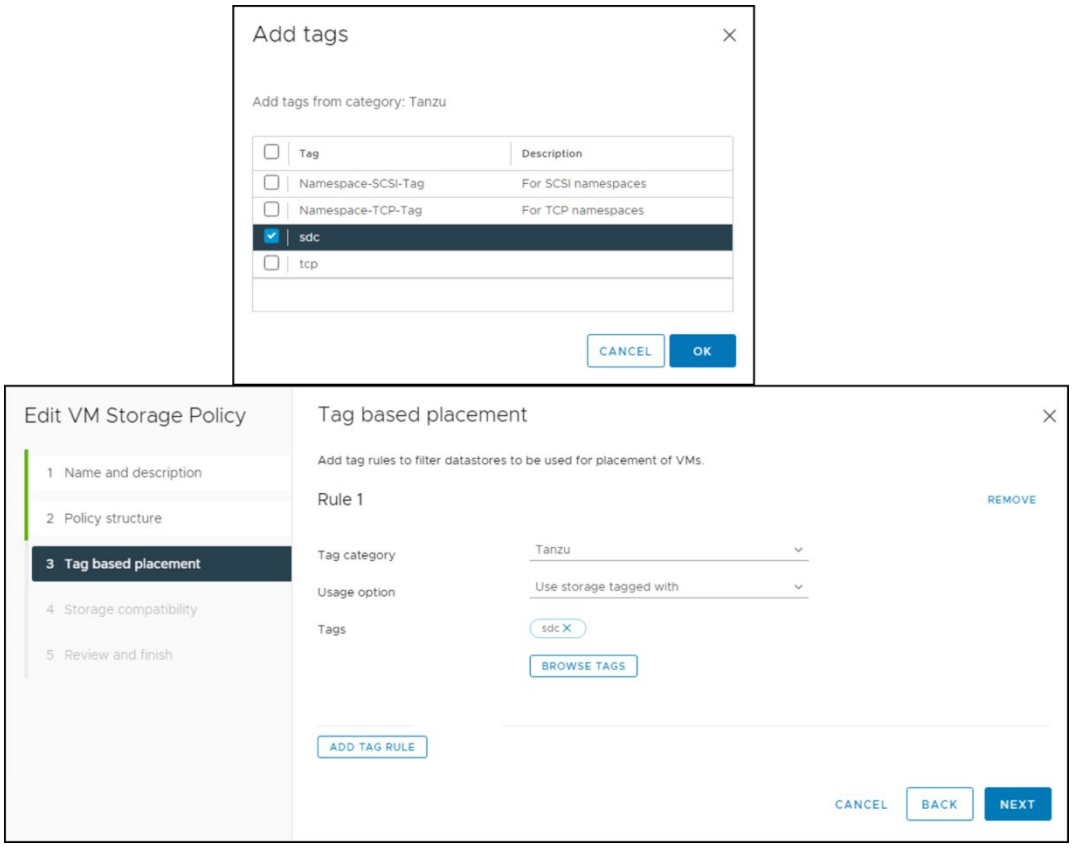

Figure 41. SDC storage policy – Step 5

12. Click **NEXT**.

With the rule applied, VMware lists any datastores that meet the assigned criteria. In this example, the SDC tag was only put on one datastore as shown in Figure 42.

Figure 42. SDC storage policy – Step 7

13. Click **NEXT** to reach the summary page, as shown in Figure 43.

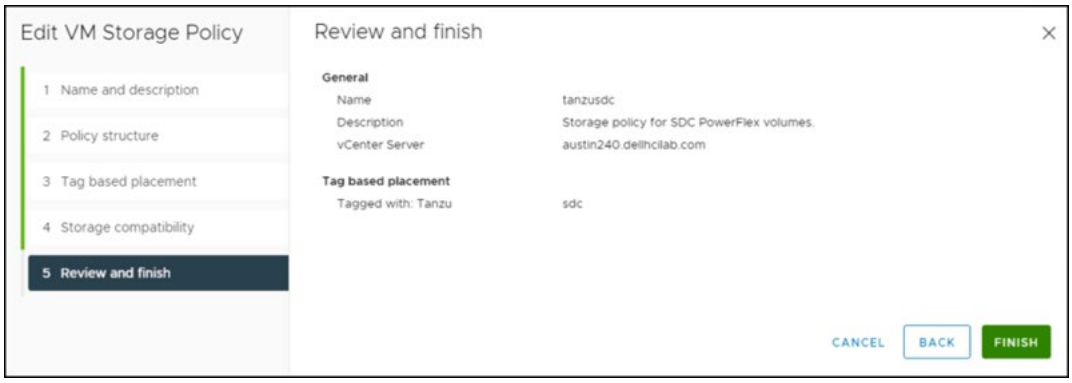

Figure 43. SDC storage policy – Step 8

14. Review the selections, and then click **FINISH**.

TCP

The TCP storage policy is created in the same way as the SDC policy, but the tag is *TCP* instead of SDC. The TCP policy is shown in Figure 44.

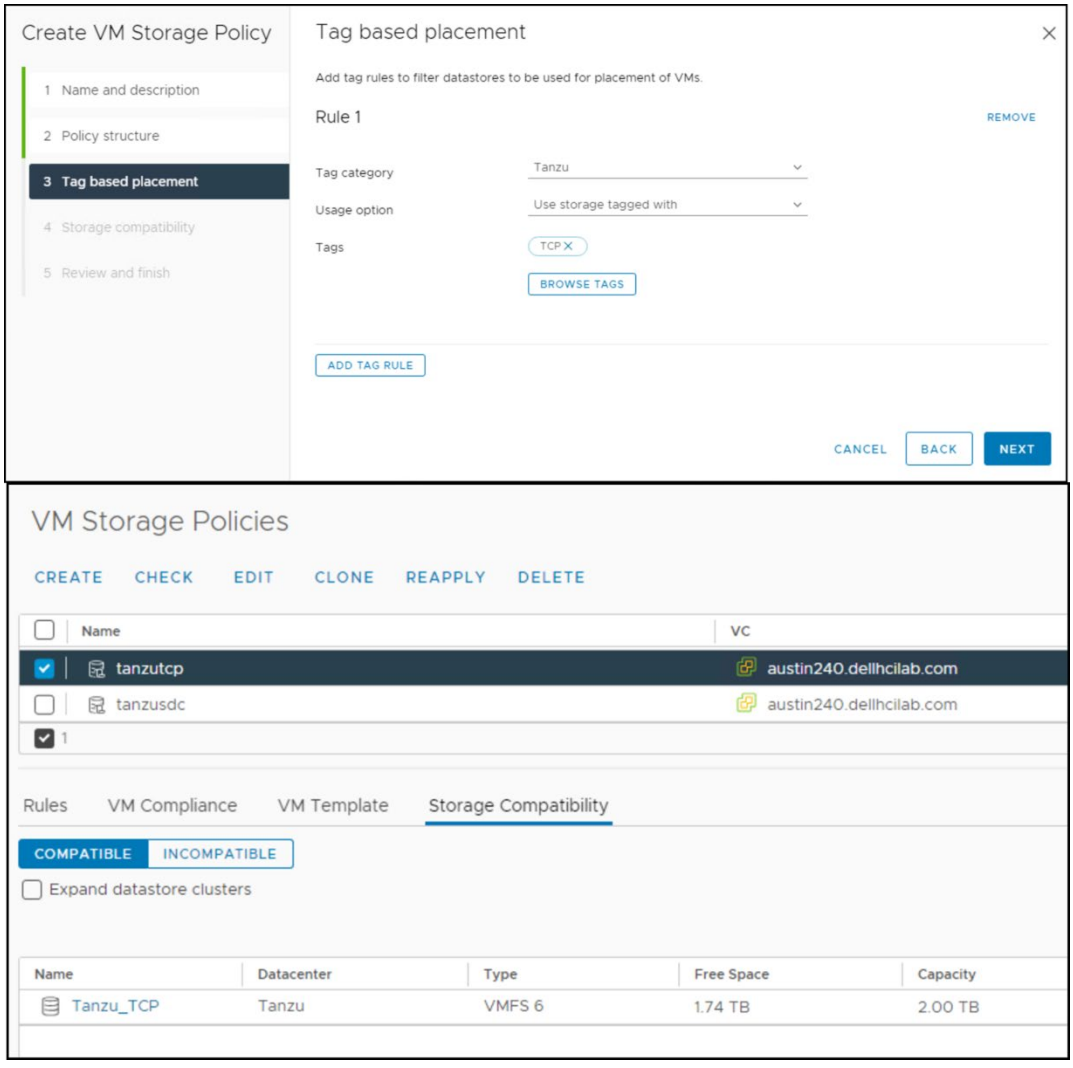

Figure 44. TCP storage policy

Namespace tags

In addition to the tags for the Tanzu infrastructure, you should create ones for the Namespaces. The Namespaces enable the administrator to set resource and object limits for the chosen users. Specific Kubernetes clusters and Storage Policies are assigned to these namespaces. To assign the storage policies, you must setup the tags and assign them to datastores.

The two tags for this Tanzu environment are shown in Figure 45 using the same **Tanzu** category. These tags are array-agnostic as this storage setup has more arrays than PowerFlex. Therefore, there are tags for:

- Any SCSI-presented storage
- **Namespace-SCSI-Tag**
- Any NVMe/TCP-presented storage (**Namespace-TCP-Tag**)

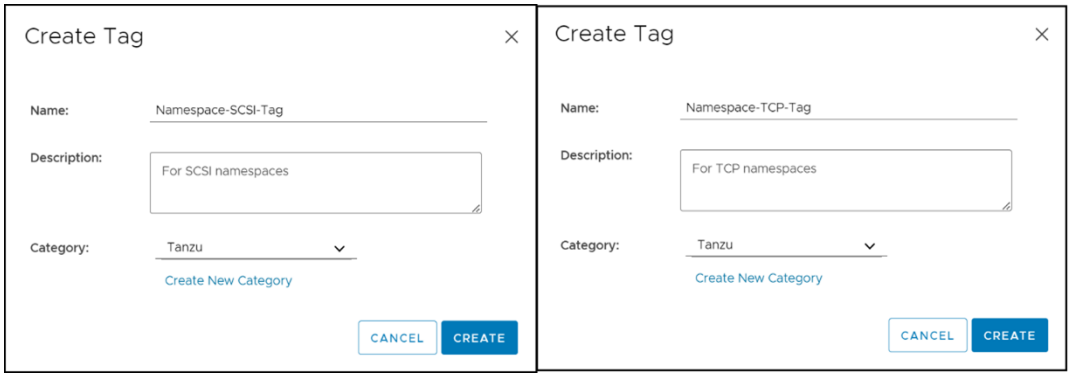

Figure 45. Namespace tags

After creating the tags, associate them with any datastores that are used for persistent volumes, as described in Provision storage.

Namespace storage policies

These tags are now used for the namespace storage policies. Since the wizard was covered previously in VMware storage policies, Figure 46 and Figure 47 show the compatible datastores (with tag applied) and the results.

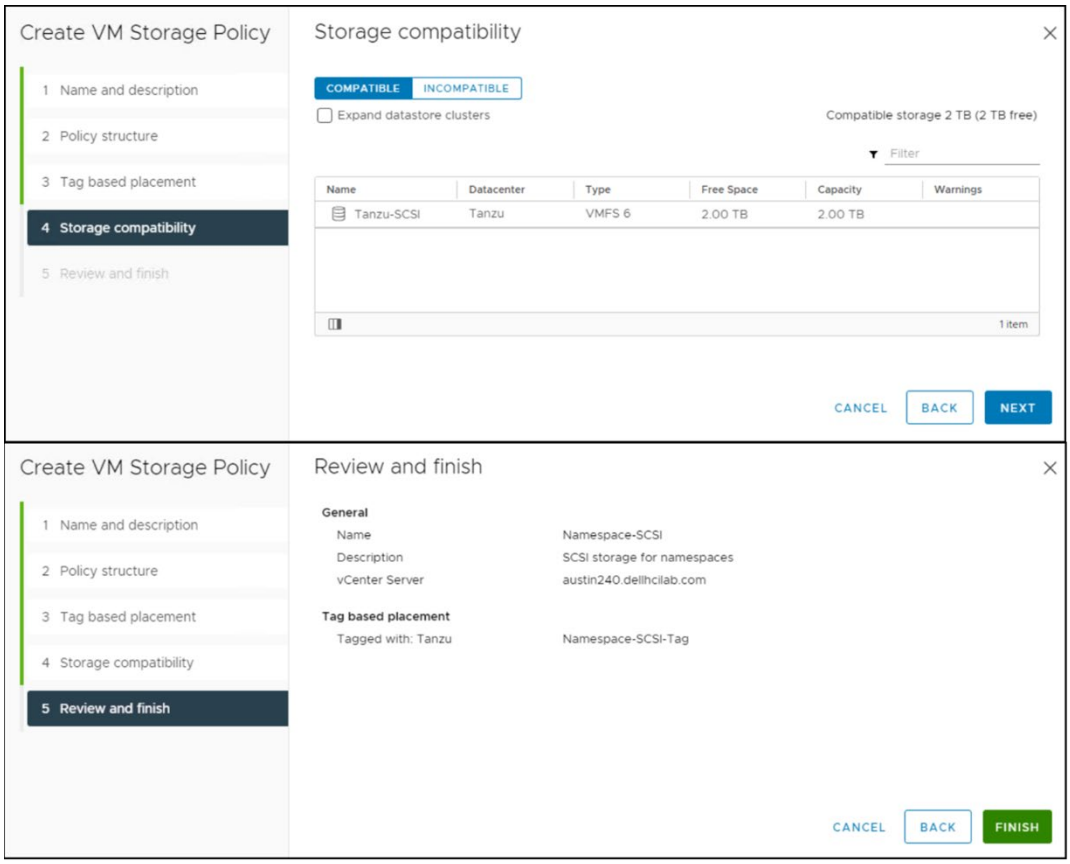

Figure 46. Storage policy Namespace – SCSI

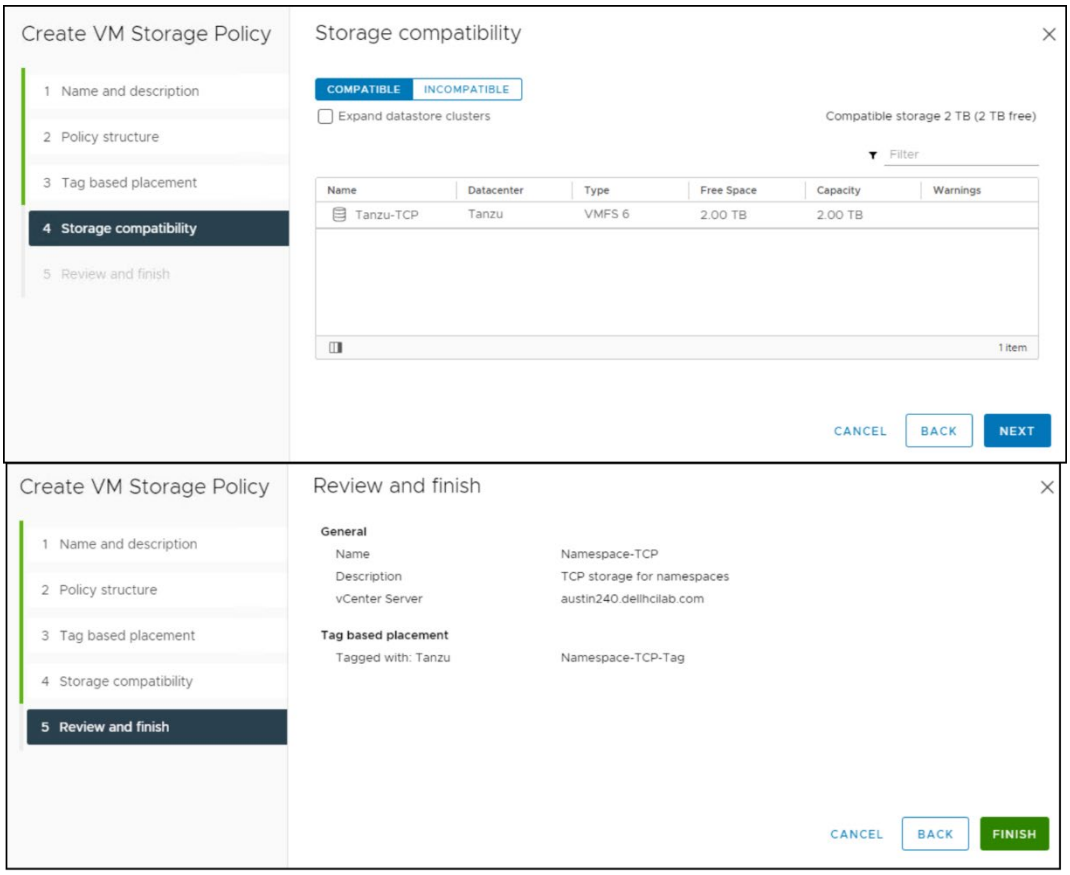

Figure 47. Storage policy Namespace – TCP

Adding a content library

Introduction

For Tanzu to create the TKG Clusters, it must have access to the online templates for the virtual machines. This access is gained through the addition of a content library. A content library is a container object for VM templates and other types of files. These files can be ISO images, text files, or any other type of file that users may want to share across a VMware environment.

A content library can be local or subscribed. A local library is one in which the VMware administrator uploads and controls access to the files through publishing. A subscribed library attaches to an existing published library to which the VMware administrator has no control. The owner of the subscribed library controls publishing.

Content library configuration

This procedure describes adding the TKG content library in preparation for the Tanzu installation.

1. In vSphere, click the **hamburger menu** ( ) on the left.
2. Go to the **Content Libraries** option. The **New Content Library** dialog window displays as shown in Figure 48.

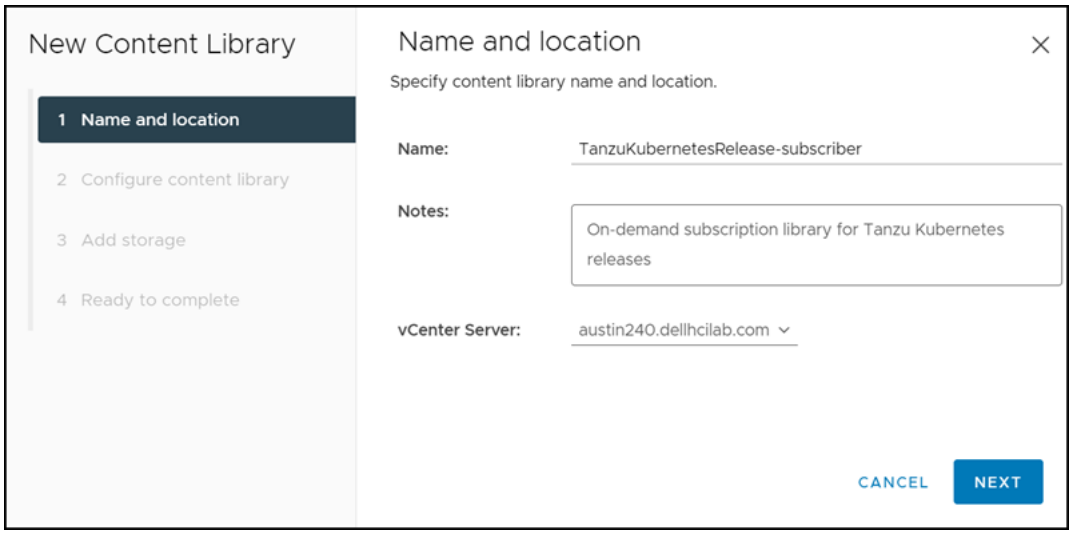

Figure 48. Add content library – Step 2

3. Enter a **Name** (required), **Notes** (optional), **and vCenter Server** if not already defaulted.
4. Click **NEXT** to advance to the **Configure content library** dialog window.

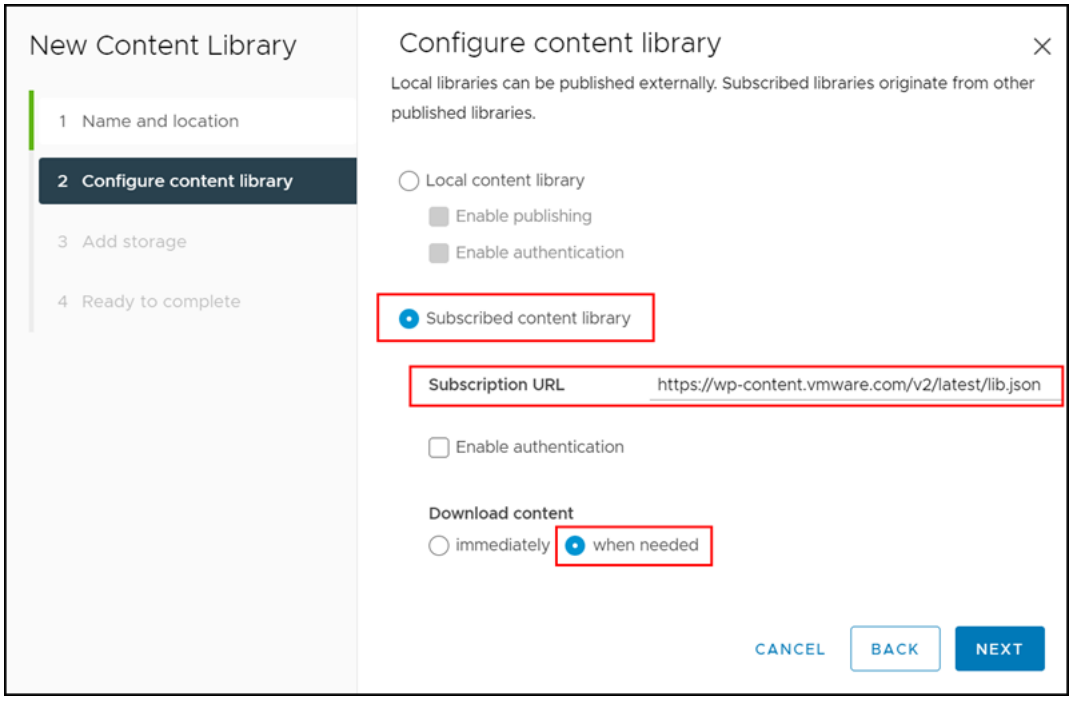

Figure 49. Add content library – Step 3

5. Select the **Subscribed content library** radio button.
   a. Enter the **Subscription URL** that VMware provides.
6. Select the **Download content -> when needed** radio button.

Note: Doing so ensures that VMware gathers only what it requires when it deploys the TKG Clusters.

---

7. Click **NEXT**. An authenticity verification pop-up dialog opens.

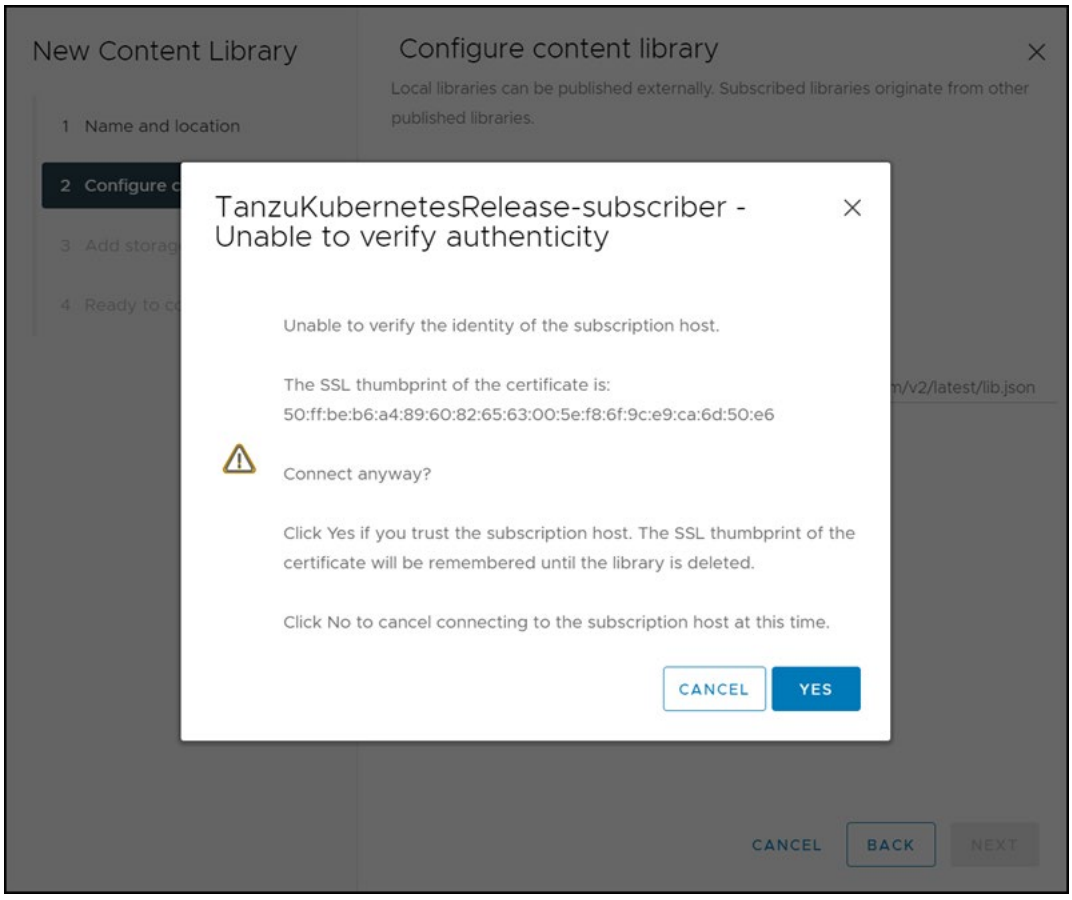

Figure 50. Add content library – Step 4

8. Click **YES** to verify the authenticity of the library.
9. In the next dialog, use the drop-down box to select **OVF default policy** as the security policy.
10. Leave the **Apply Security Policy** checkbox selected.
11. Click **NEXT** to advance to the **Add storage** dialog window as shown in Figure 51.

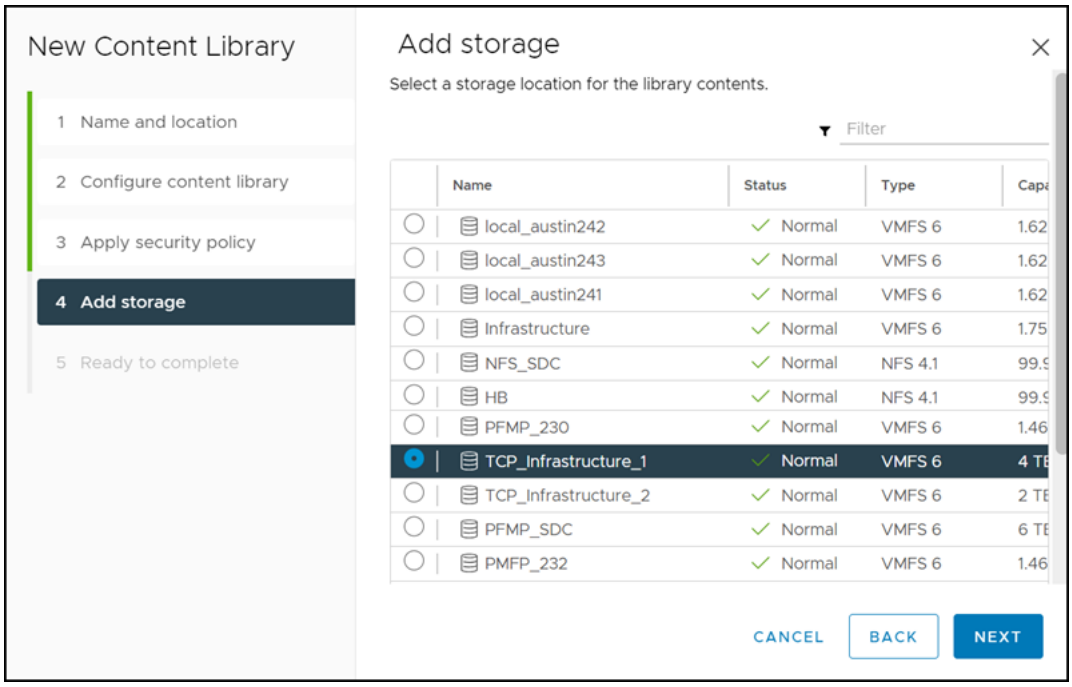

Figure 51. Add content library – Step 6

12. Choose a **datastore** where the library content is stored.

Note: Since the images are on-demand, the storage usage should be minimal. The type of datastore is not relevant. In this example, a PowerFlex datastore that is presented using NVMe/TCP is selected.

13. Select **NEXT** to continue.
14. Review all the settings in the summary page, and then click **FINISH**.

The library is now added and available.

Chapter 6 vSphere with Tanzu implementation

This chapter presents the following topics:

Introduction

Although Tanzu is integrated into vSphere, as explained earlier, it still requires some external network component. Therefore, before starting the Tanzu implementation within vCenter, the user must either install NSX-T, NSX Load Balancer, or HA Proxy. This environment uses the NSX Load Balancer.

NSX load balancer

Since there are different networking options the user may choose, this guide does not go into the detailed installation of the NSX Load Balancer. Both HA Proxy and NSX Load Balancer enable most of the Tanzu functionality that customers use, while NSX-T is a complete solution.

Installing the NSX Load Balancer is straightforward. It is deployed as a single VM using an OVA file. The single VM, once deployed, can be expanded into a multinode cluster for production environments as was the case in this solution. The three nodes are shown in Figure 52.

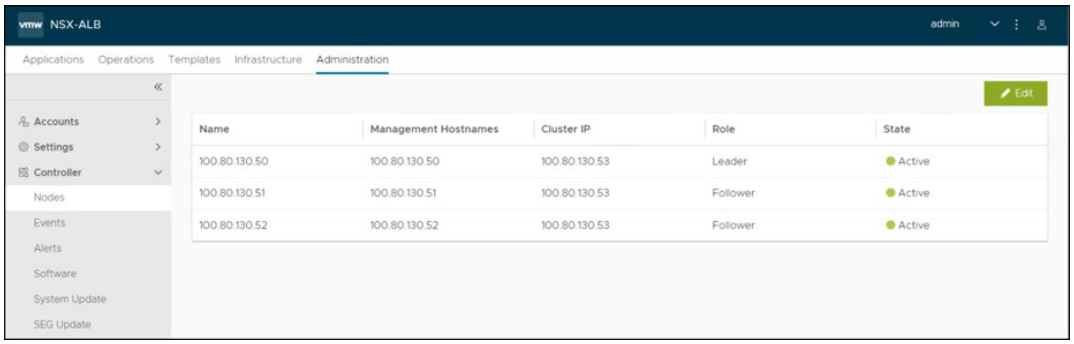

Figure 52. Multinode NSX-LB implementation

VMware offers step-by-step instructions for configuring the NSX Load Balancer in preparation for use with Tanzu. These instructions were followed for this guide:

https://docs.vmware.com/en/VMware-vSphere/7.0/vmware-vsphere-with-tanzu/GUID-A51FAF35-D604-4883-A93D-58463B404C4E.html

Tanzu

With all prerequisites completed including the NSX Load Balancer installation, users can configure Tanzu.

vSphere with Tanzu is not truly enabled until a Supervisor Cluster is installed. The Supervisor Cluster is at the heart of Tanzu, providing the necessary resources for the environment. A Supervisor Cluster is created with the Workload Management screen of the vSphere Client. It is associated directly with a cluster within vSphere which has at least two ESXi hosts. The Supervisor Cluster, once created, enables the user to run Kubernetes workloads. VMware portrays the architecture in the following simple diagram in Figure 53.

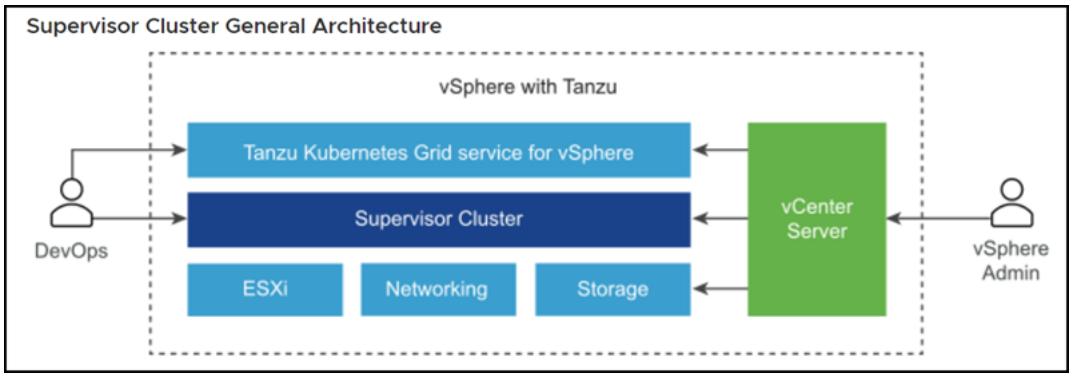

Figure 53. Supervisor Cluster

ESXi houses additional components for Tanzu, as detailed in Figure 54. There is a Spherelet which is a kubelet that is ported to ESXi, which enables the host to be part of a Kubernetes cluster. Three Kubernetes control planes are also present in total, even if there are only two ESXi hosts that make up a Kubernetes cluster. The CRX, or Container Runtime Executive, is for vSphere Pods. Finally, not listed here are the essential two modules which makes Kubernetes cluster provisioning possible: The Cluster API and VMware Tanzu Kubernetes Grid Service.

Note: Spherelets and the vSphere Pods are not supported with vSphere Networking as used in this guide.

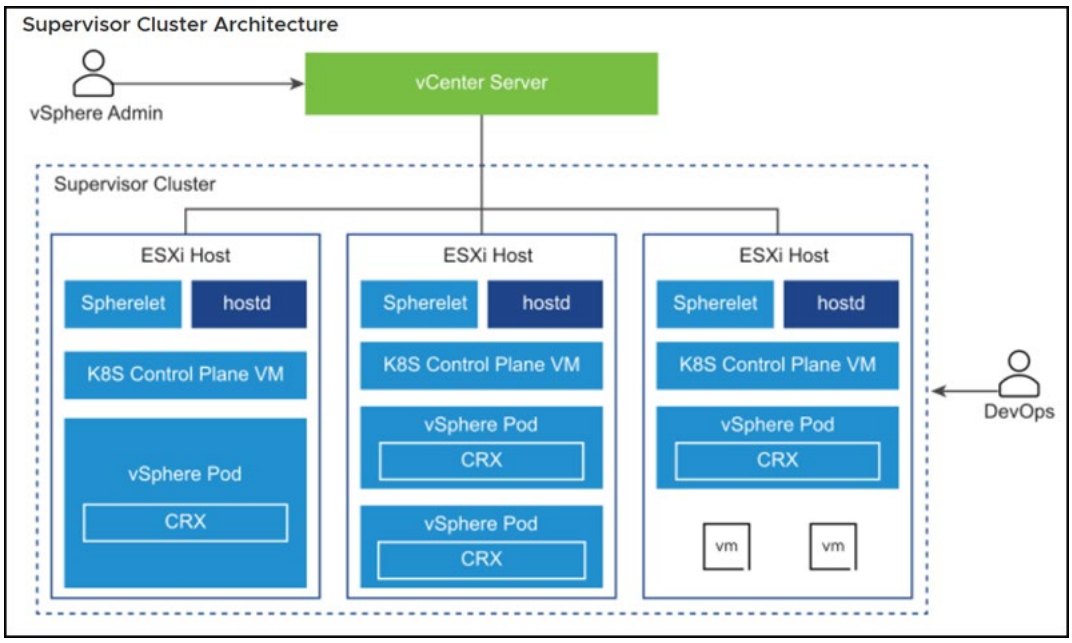

Figure 54. Supervisor Cluster ESXi detail

Finally, VMware has three different ways that the Tanzu Kubernetes Grid helps the user to manage the environment. At a high level, these methods are:

- **Tanzu Kubernetes Grid Service**—It is what provisions the clusters. It includes a Cloud Provider Plugin (NCP in diagram). Most importantly, the Kubernetes cluster passes requests for persistent volumes (for example, on PowerFlex datastores) to

the Supervisor Cluster. This service is integrated with VMware Cloud Native Storage (CNS).

- **Cluster API**—Developers can use a tool more familiar to them to accomplish the same as what the UI would offer.
- **Virtual Machine Service**—It manages the life cycle of the control plane and worker node VMs hosting a Tanzu Kubernetes cluster.

These components are displayed in Figure 55.

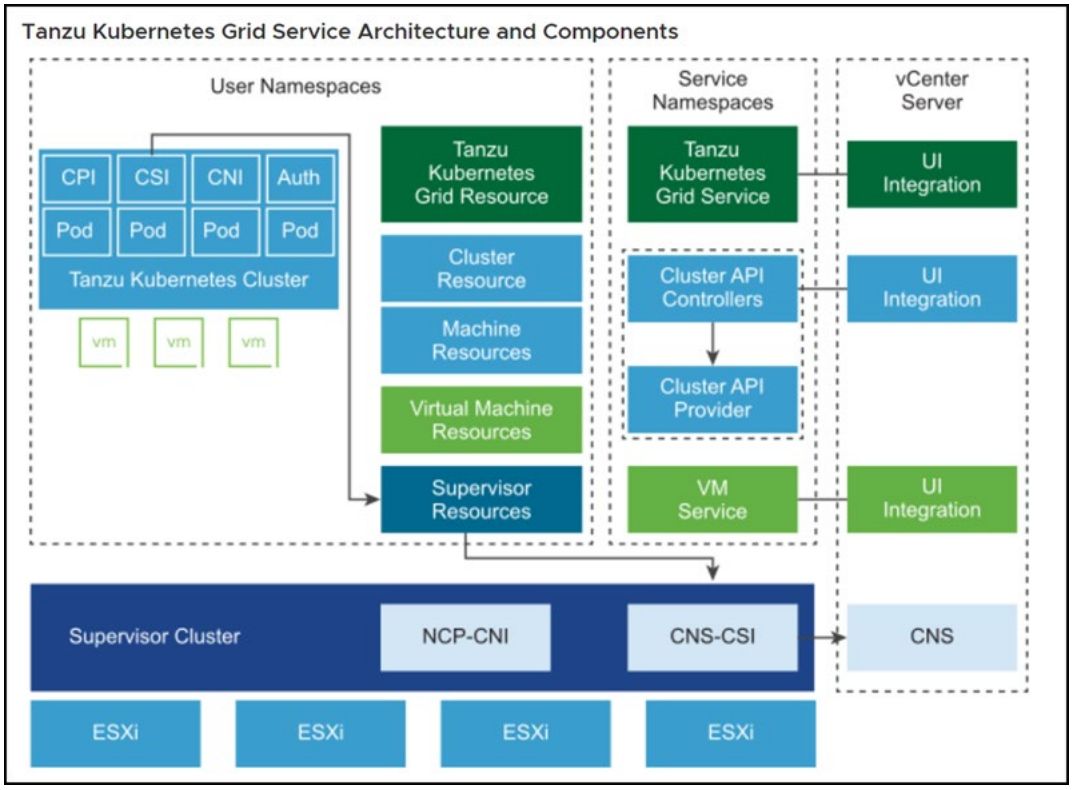

Figure 55. Tanzu Kubernetes Grid Service architecture

CNI and CNS

Storage is essential for using PowerFlex with Tanzu. This section provides a detailed discussion about the Container Storage Interface (CSI) and Cloud Native Storage (CNS).

CNS enables customers to consume persistent storage in Kubernetes, while providing automation or life cycle management. VMware offers the infographic in Figure 56, which shows the CNS components. VMware uses a CSI, or Container Storage Interface, to provision storage from various sources. A CSI is an implementation of the CSI specification which dictates how to connect storage to the orchestration tools in containers.

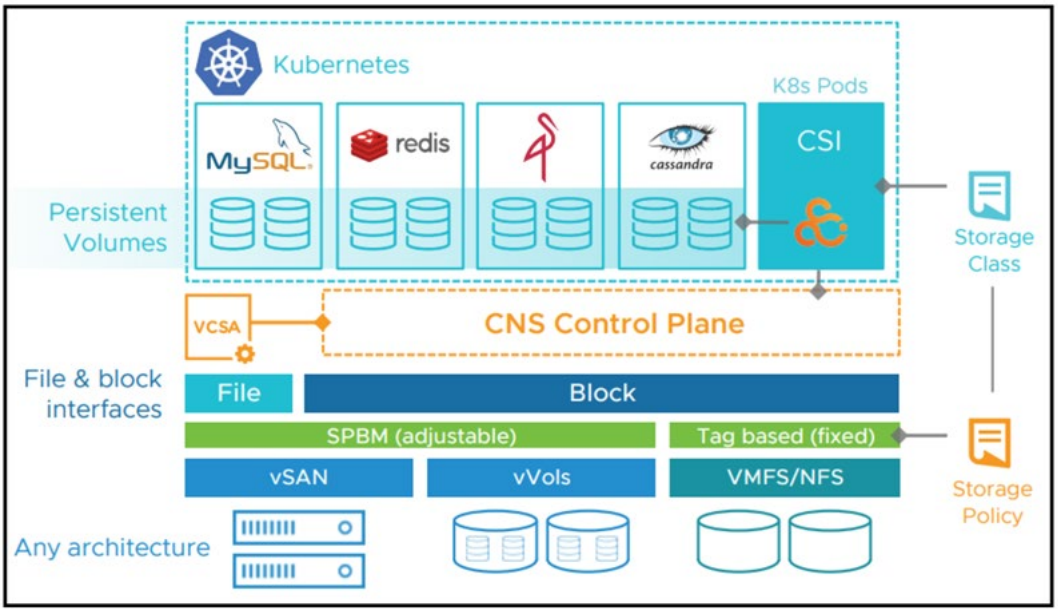

Figure 56. CNS and CSI in vSphere

The vSphere CSI uses Storage Policy Based Management (SPBM), or storage policies with storage classes, to create persistent storage for Kubernetes. As shown in the diagram, that storage could be anything from vSAN to vVols, VMFS to NFS.

First Class Disks

Volumes that are created with vSphere CSI are known as First Class Disks, or FCDs. What makes FCDs special is that unlike most disks in vSphere, these disks do not have to be associated with a VM. A .vmdk in vCenter is considered based on the VM to which it is attached.

Through editing the VM users can expand the disk, delete it, or even temporarily remove it, and add it to another VM. However, its life is associated with another object, unlike with FCDs. A First Class Disk has its own life cycle which can be controlled outside of the VM. Users can expand it, snapshot it, or delete it without it ever being tied to a VM.

Cloud Native Storage

Although VMware presents CNS as all the components in Figure 56, it really has two parts:

- A vSphere volume driver for Kubernetes (vSphere CSI)
- The integration into vCenter shown in Figure 57

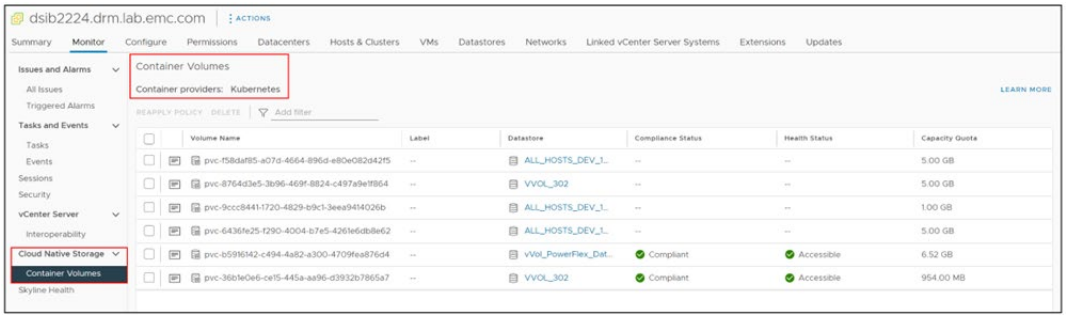

Figure 57. CNS vCenter integration

The volume driver consists of two pieces; a syncer and the vSphere Container Storage Interface (CSI) driver. CNS controls all aspects of volume provisioning with the storage control plane. It handles the entire life cycle – creation, deletion, and so forth, including snapshots, clones, and general health. One concept that is essential to this model is that these volumes are independent of the life cycle of VMs. This independence is only possible with the FCDs.

Storage vendor CSIs

While the vSphere CSI does not know or care about the underlying storage, vendor CSIs are designed to provision only from their own storage. By design, these CSIs are not part of the VMware CNS. However, they can co-exist in the same Kubernetes environment. With the vSphere CSI, it is not possible to provision a device directly from say the PowerFlex array to a cloud-native application.

Any vSphere CSI-provisioned device from PowerFlex 4.0.x is a .vmdk in a VMFS or NFS datastore. Tanzu does not permit implementation of vendor CSI drivers on the Supervisor Cluster. It may be technically possible to use a vendor CSI driver on a Tanzu Kubernetes cluster. However, VMware would not support the steps that would be required to do so. VMware locks down the virtual machines that make up the cluster as seen in Figure 58.

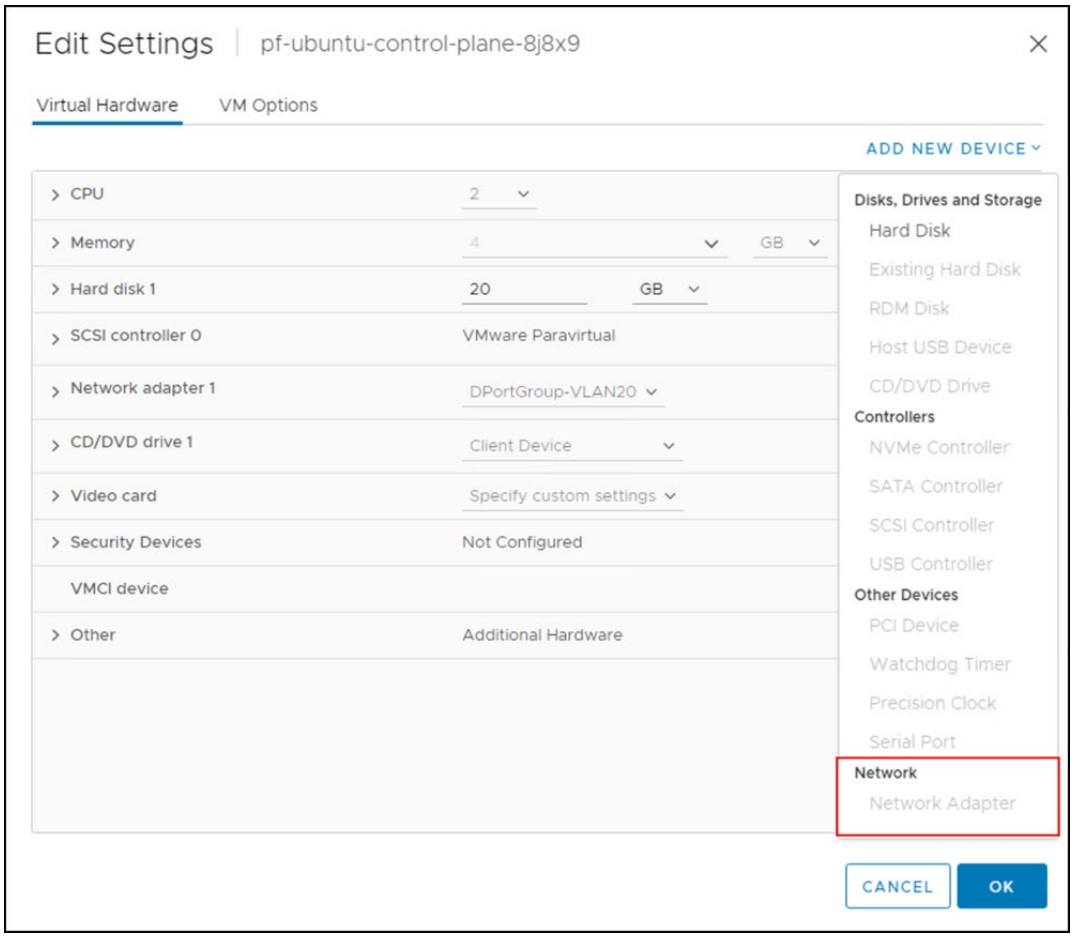

Figure 58. Tanzu Kubernetes cluster node

Users must add network interface controllers and gain access to the operating system to install the SDC driver to use the PowerFlex CSI. NVMe/TCP is not supported with the

CSI. Since the VM is purposely locked-down by VMware, anything that is done to unlock it, by definition, would be unsupported. Dell Technologies advises customers that want to use the PowerFlex CSI to install Kubernetes on VMs that are not part of the Tanzu environment.

Configure workload management

The Supervisor Cluster is configured within vCenter. All steps and images are included as the wizard is likely new for most users.

1. In vSphere, click the **hamburger menu** ( ) on the left.
2. Go to the **Workload Management** option.
3. Select the **Supervisor Clusters** tab, and then click **ADD CLUSTER**.

Figure 59 shows an existing Supervisor Cluster, named *Management*. This example adds another Supervisor Cluster. Each Supervisor Cluster is associated with a vSphere Cluster of at least two ESXi hosts.

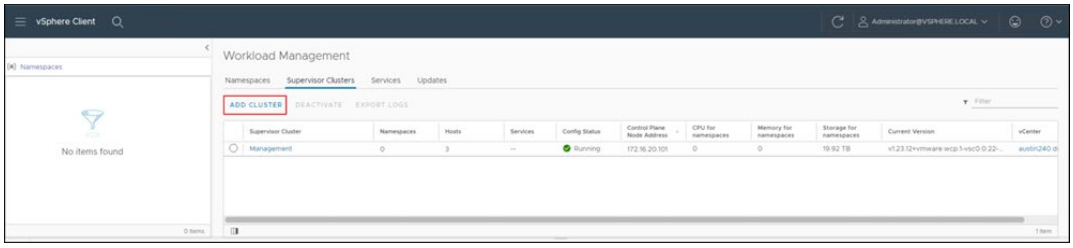

Figure 59. Workload Management – Add Cluster

If there is only a single vCenter it defaults automatically, as shown in Figure 60.

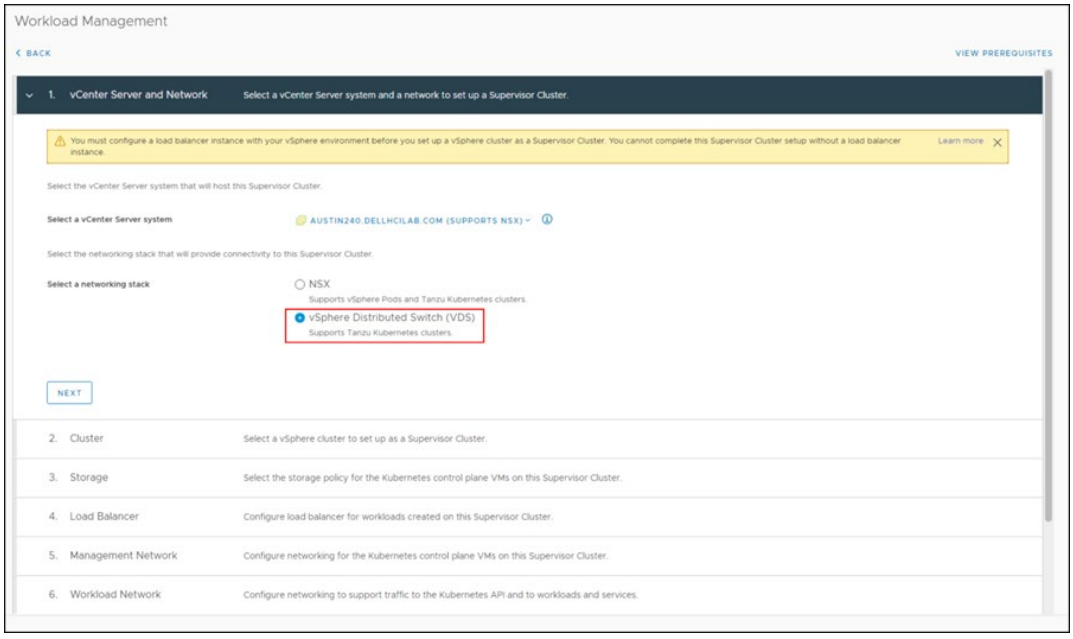

Figure 60. Workload Management – Step 1

4. Select the networking stack, as shown in Figure 61.

Note: vSphere Networking is being used in this example. For VDS, a load balancer must already be installed. The wizard requires users to indicate which one in a later step.

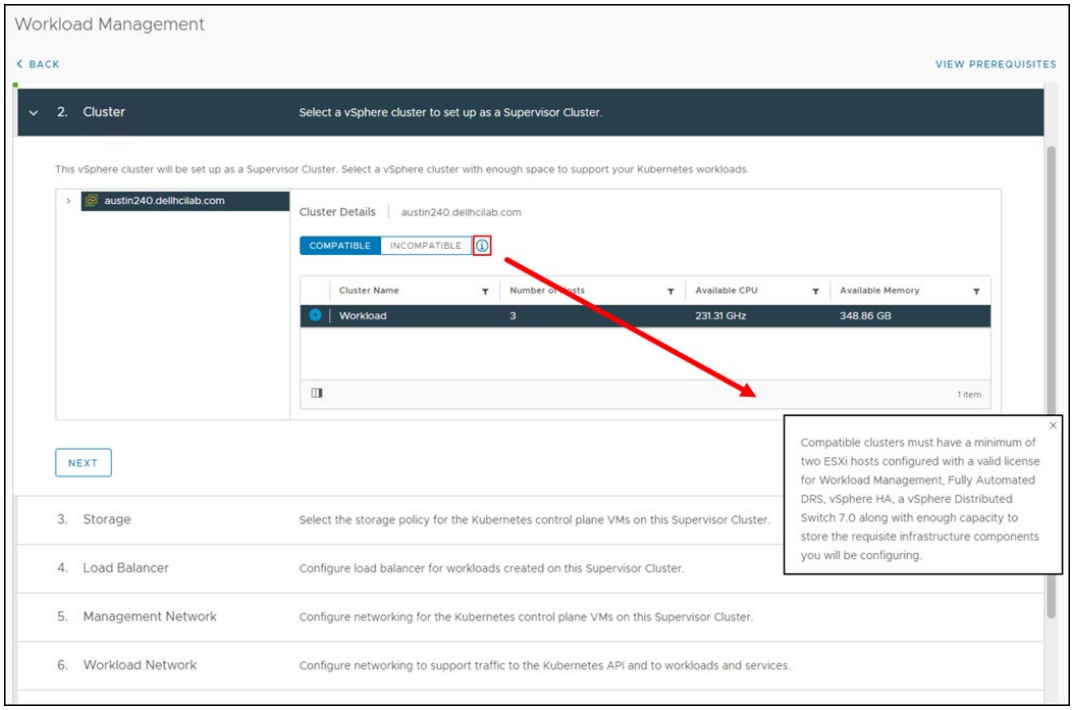

Figure 61. Workload Management – Step 2

5. Select a cluster. In this example, there is one remaining cluster with three ESXi hosts.
   a. Select the **information bubble** next to the **COMPATIBLE** button. VMware displays a warning that a minimum of two ESXi hosts are required to create a Supervisor Cluster.
6. Use the **drop-down box** to select a previously created storage policy, either for SDC or NVMe/TCP (as covered in VMware storage policies), as shown in Figure 62.

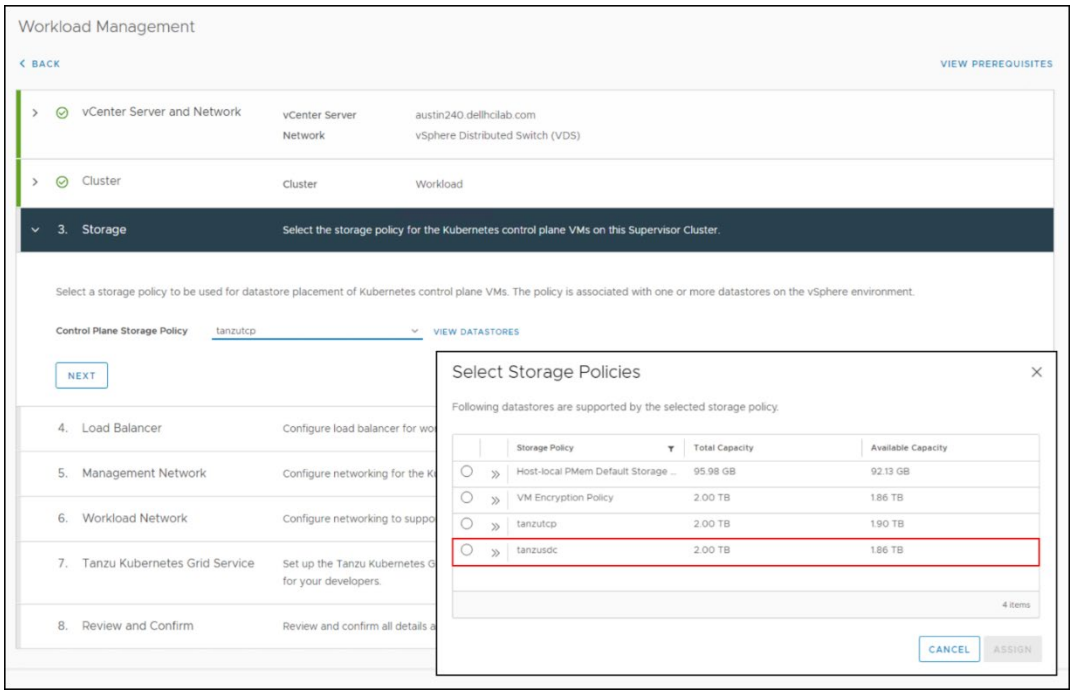

Figure 62. Workload Management – Step 3

Figure 63 indicates that all files that are related to this Supervisor Cluster should be stored in datastores that are tagged with the **TCP** tag.

7. Select the **radio button** next to the policy. In this example, the **tanzutcp** storage policy is selected.
8. Click **ASSIGN** to progress to the Load Balancer dialog**.**

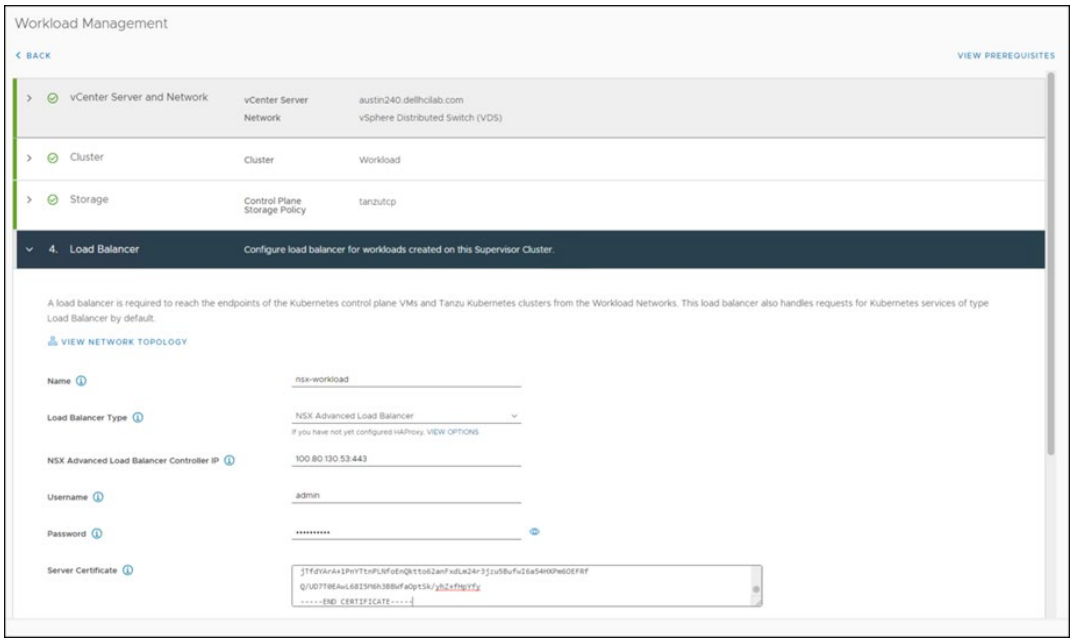

Figure 63. Workload Management – Step 4

The previously created NSX Advanced Load Balancer (ALB) information is required for this step.

9. Provide a **Name** for the ALB.

Note: The name is only an identifier. It is not intended to resolve to an IP, for example.

10. Select **ALB** for the Load Balancer Type.
11. Enter the **controller IP address**. If the ALB is a cluster, use the VIP that was configured when the ALB was installed.
12. Enter a **Username** and **Password**.
13. Paste the **Server Certificate** into the box that was generated in the ALB.

Note: The certificate information is available within the NSX-ALB interface under **Templates -> Security -> SSL/TLS Certificates**.

14. Expand the Management Network dialog, as shown in Figure 64.

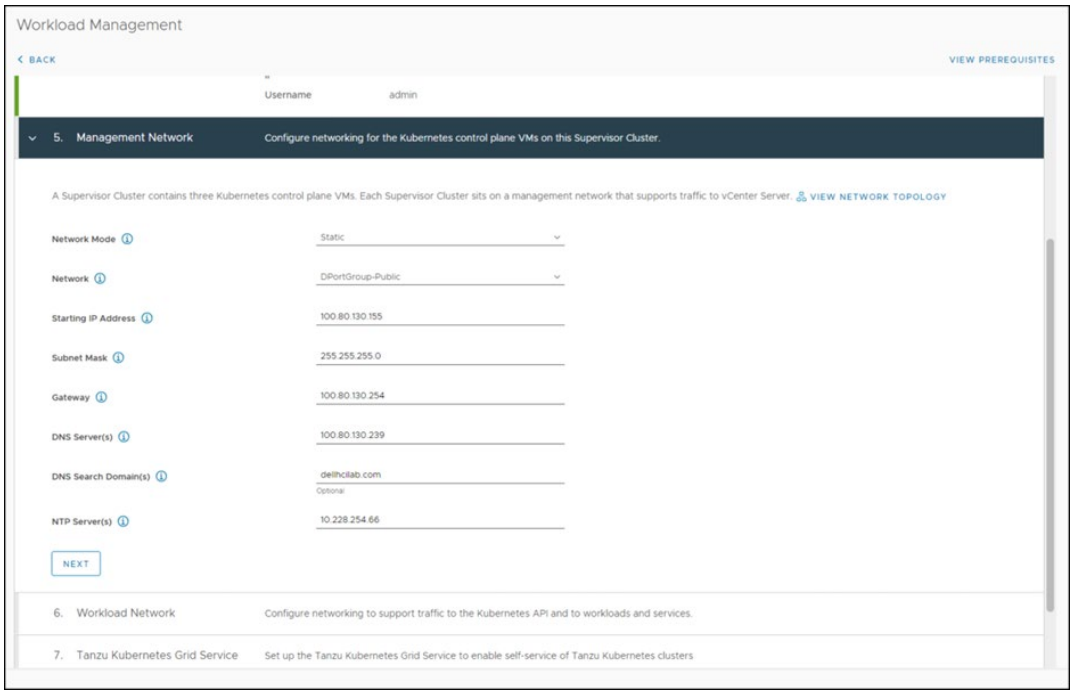

Figure 64. Workload Management – Step 5

15. Input the **Management Network** by selecting either DHCP or Static under **Network Mode**. In this example, a static configuration is used which necessitates entering all the fields.
    a. Since a Supervisor Cluster is three nodes, the provided **Starting IP Address** is the first node.
    b. The second and third IP addresses are the following two IPs in sequence.
16. Expand the **Workload Network**, as shown in Figure 65.

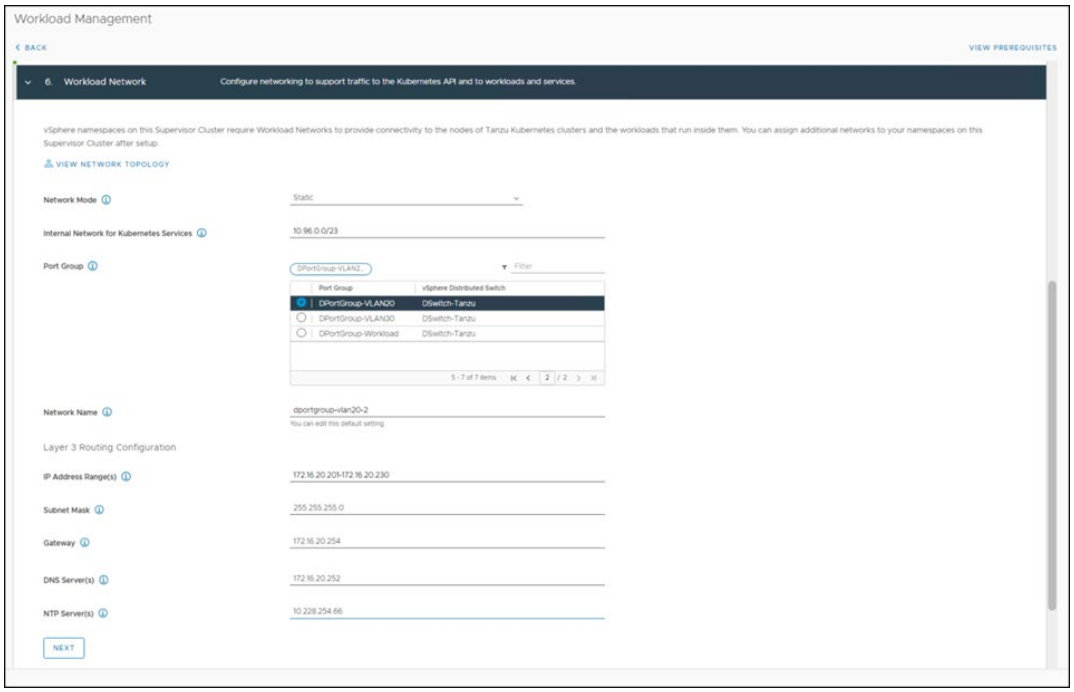

Figure 65. Workload Management – Step 6

The Workload Network is used to communicate with future Kubernetes clusters. Generally, this network is an internal network running on its own VLAN, but it must be able to route to the Management Network. Both static and DHCP routes are supported for the Network Mode.

17. Select the **Network Mode**.
18. If using static mode, the CIDR for the **Internal Network for Kubernetes Services** can be left as default.
19. Enter a **Network Name**. As with ALB, the name is only an identifier.

Note: Be sure to supply enough IP addresses in the **IP Address Range** to satisfy the intended Kubernetes cluster or clusters.

---

20. Click **NEXT** to progress to the **Content Library** dialog, as shown in Figure 66.

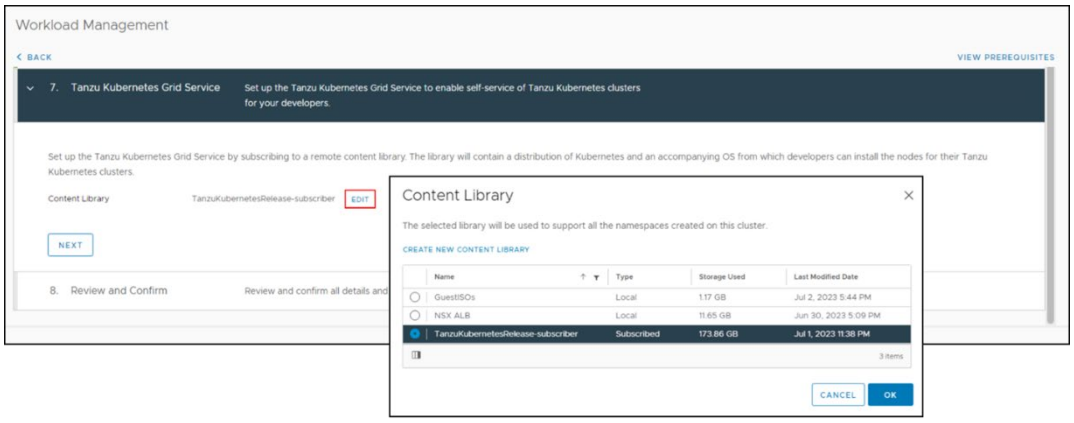

Figure 66. Workload Management – Step 7

21. Add the content library that was created in the section Adding a content library.

Doing so enables the Supervisor Cluster to deploy Kubernetes clusters.

22. Click **OK**.
23. Click **NEXT** to progress to the **Review and Confirm** dialog, as shown in Figure 67.

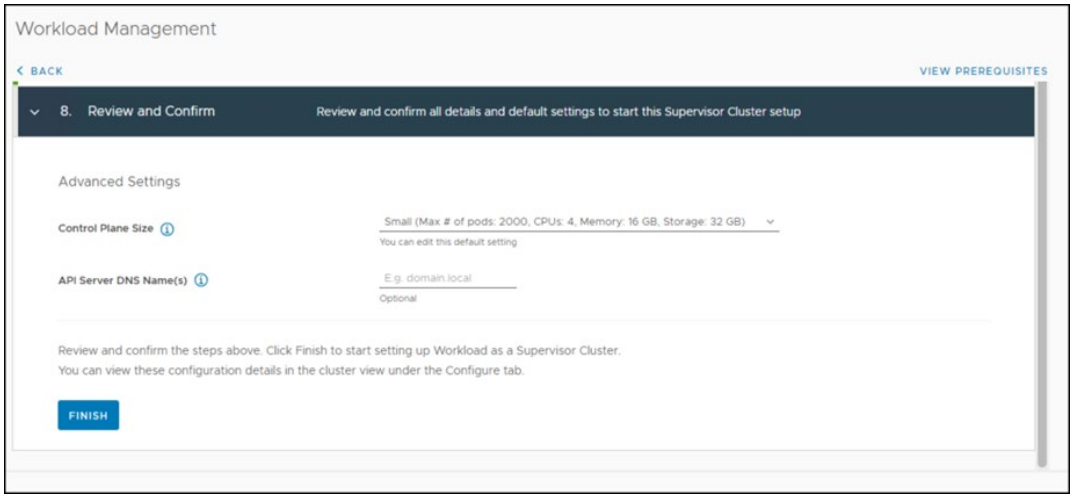

Figure 67. Workload Management – Step 8

24. Ensure that all information is correct before proceeding.

Note: If changes are required, then **each step in this section must be repeated**. You cannot update a single item.

---

25. Once all information is confirmed, click **FINISH**.
26. While the cluster is under configuration, you can review them by selecting the hyperlink (for example, *1*) in the Config Status column. The output is shown in Figure 68.

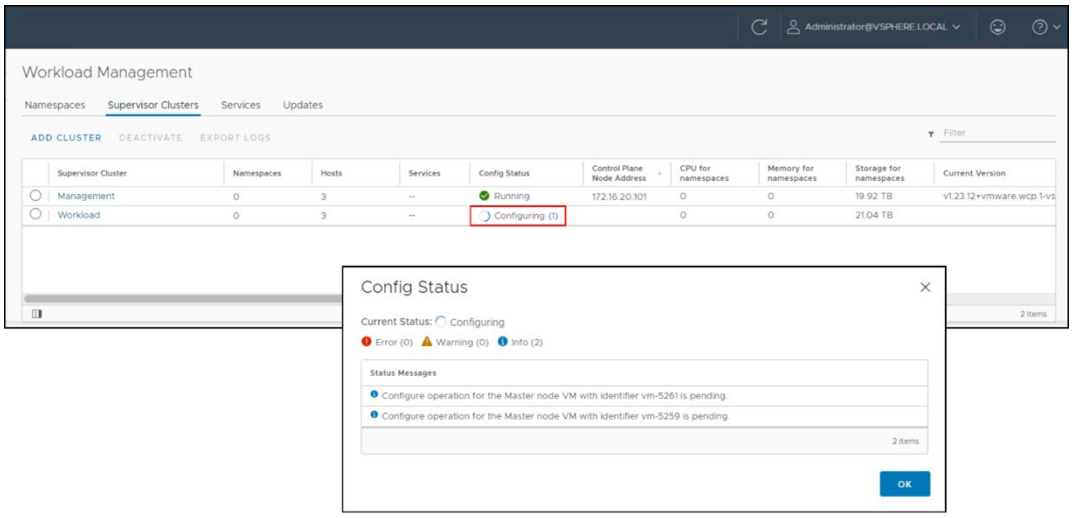

Figure 68. Workload Management – Config Status

The implementation of the Supervisor Cluster can take some time to complete. The **Config Status** changes to **Running** when it is done.

Namespace

After creating the Supervisor Cluster, you can create a namespace to manage resources. VMware creates this namespace on the Supervisor Cluster (for example, *kubectl create namespace*). The VMware namespace enables the administrator to control the resources of the Kubernetes cluster environment.

1. In vSphere, go to the **Workload** cluster and the **Namespace** tab, as shown in Figure 69.

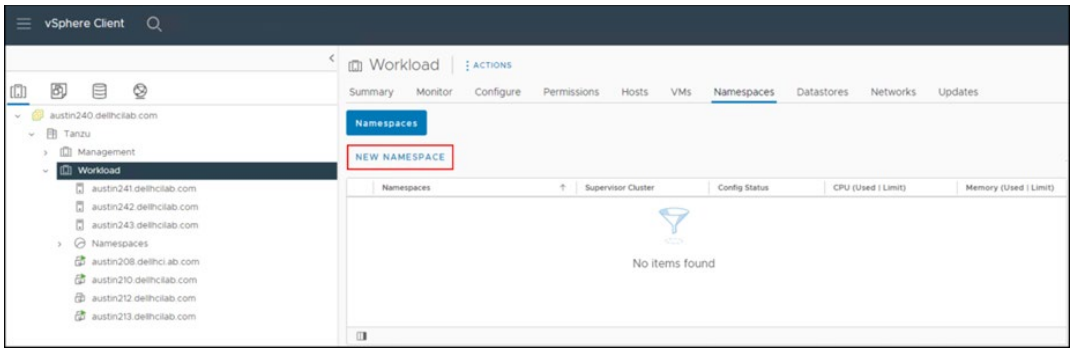

Figure 69. Create namespace – Step 1

2. Click **NEW NAMESPACE**.

Note: This navigation path is one of many possible paths.

3. Select the **cluster** where the namespace should be created. Despite highlighting the cluster, VMware does not default to it. You must select it.
4. Enter a **Name** (required) and a **Description** (optional) for the namespace.
5. Use the drop-down box next to **Network** to select the appropriate option.

In this example, there is only a single network. If there are multiple workload networks in ALB, they are available as options. See Figure 70.

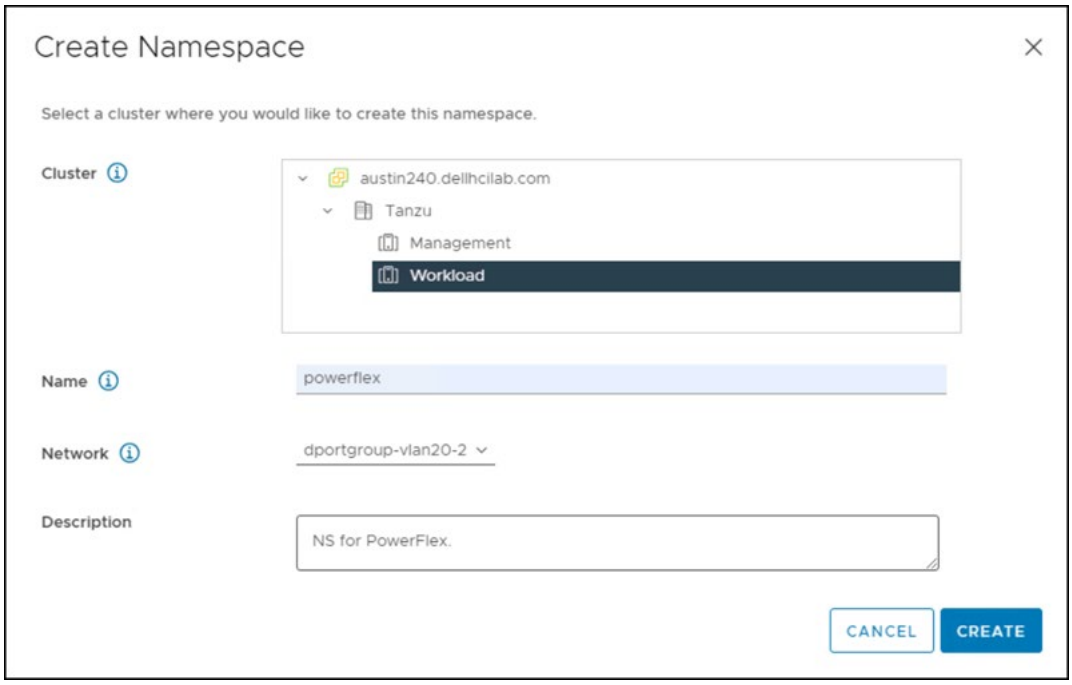

Figure 70. Create namespace – Step 2

The new namespace is immediately available. There are six different sections:

- Status
- Permissions
- Storage
- Capacity and Usage
- Tanzu Kubernetes Grid Service
- VM Service

You can modify many of these categories. Some are covered below.

Note: In the **Tanzu Kubernetes Grid Service** box, the **Content Library** is already defined, inheriting the value from the Supervisor Cluster.

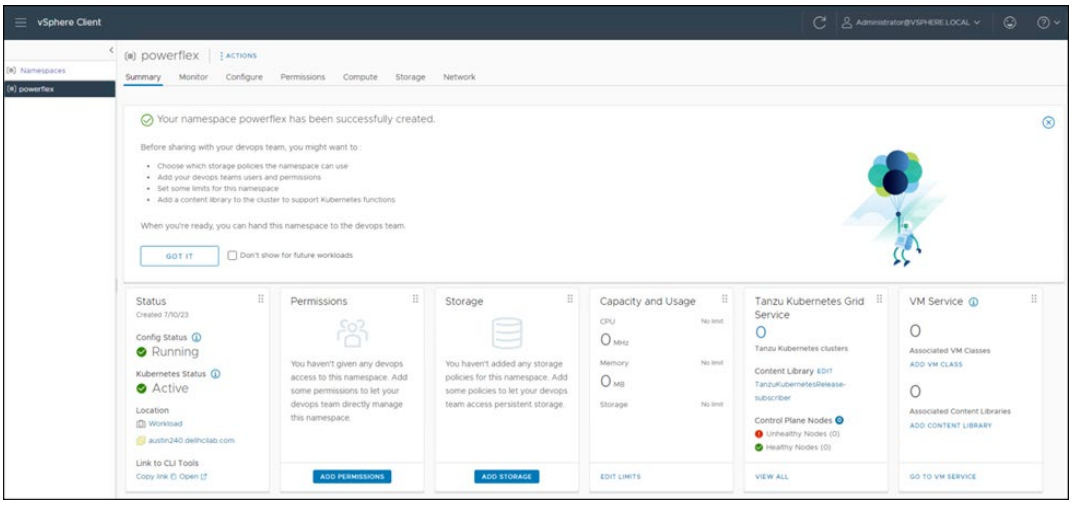

Figure 71. PowerFlex namespace

Permissions

1. Provide access to the namespace by clicking **ADD PERMISSIONS** as shown in Figure 71 above. The Add Permissions dialog appears.
2. Grant users or groups permissions to view, edit, or ownership to the namespace.

For example, if a user wants to grant view access to k8 developers from the local domain, the input appears like that shown in Figure 72.

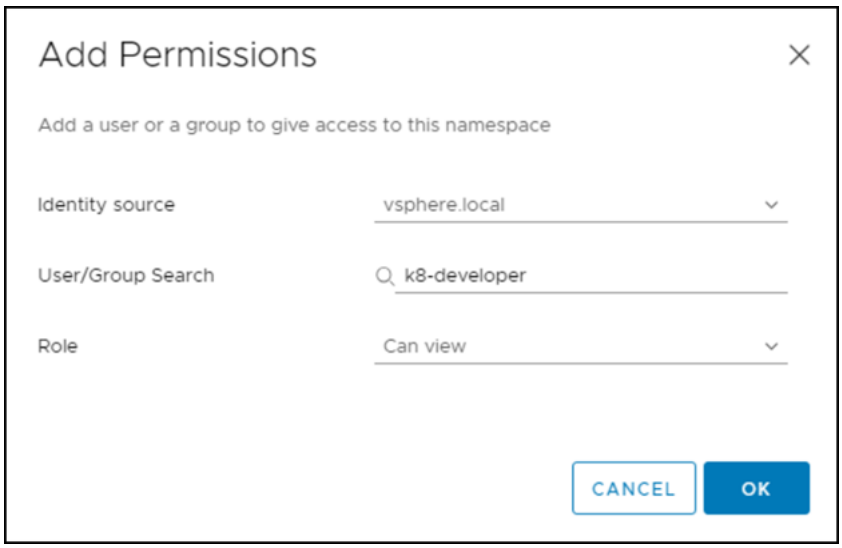

Figure 72. Namespace permissions

3. Add all necessary permissions for the namespace before continuing.

Storage

Storage was tagged for the namespaces in the section Namespace tags.

1. Provide storage policies for the namespace by clicking **ADD STORAGE** in the PowerFlex Namespace window (Figure 71 above). The Add Storage dialog appears.

2. Select the previously created storage policies - **Namespace-TCP, and Namespace-SCSI -** in Figure 73.

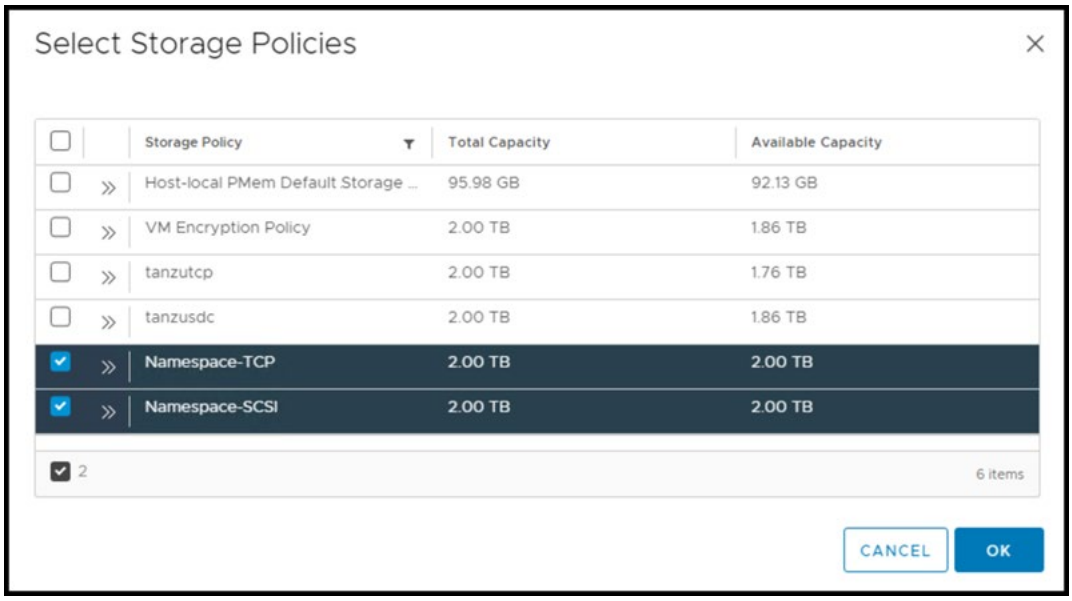

Figure 73. Namespace storage – storage policies

These storage policies are converted to Kubernetes storage classes in the Supervisor Cluster. If you want to give Kubernetes access to any other storage presented to the vCenter, it should be assigned here. Storage policies are the only way to do this access.

3. Create any additional storage policies which map to that storage and assign them here. VMware creates the storage classes.

The two storage policies in Figure 73 can be seen during Kubernetes cluster creation.

Capacity and usage

1. Provide access to the namespace by clicking **EDIT Limits** in the **Capacity and Usage** box in the PowerFlex Namespace window (Figure 71 above).
2. Optionally, enter the limitations that you want for **CPU**, **Memory**, and **Storage**. See Figure 74.

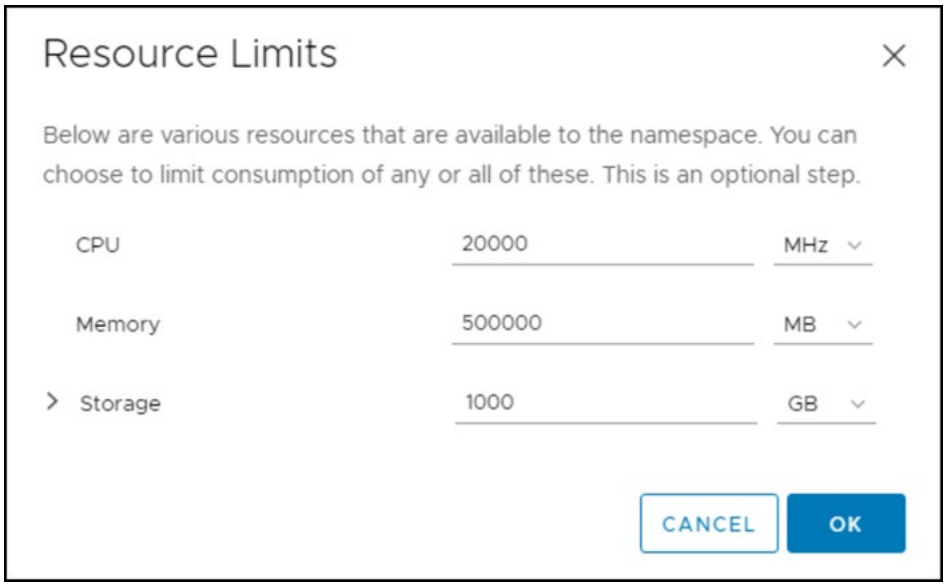

Figure 74. Namespace capacity and usage – Edit limits

VM Service

Finally, you must add some **VM Classes** which are used when creating Kubernetes clusters in the namespace. This step is mandatory.

1. Click **ADD VM CLASS** in the PowerFlex Namespace window (Figure 71 above) to display the Manage VM Classes window, as shown in Figure 75.

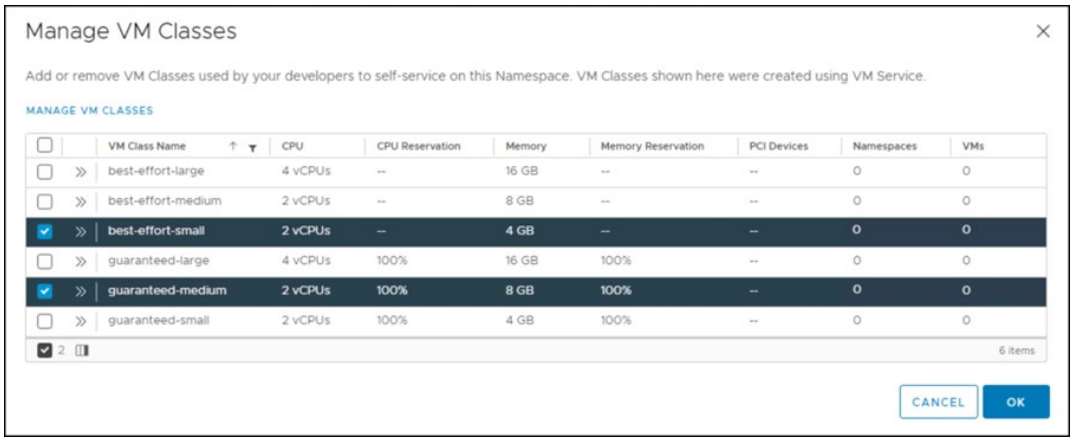

Figure 75. Manage VM Classes

2. Use the **checkboxes** to select which VMs Classes or templates to assign to the namespaces.

Note: When building a YAML file for a Kubernetes cluster, the class parameter refers to one of these classes to create the VMs.

---

3. Review the Modifications as shown in Figure 76.

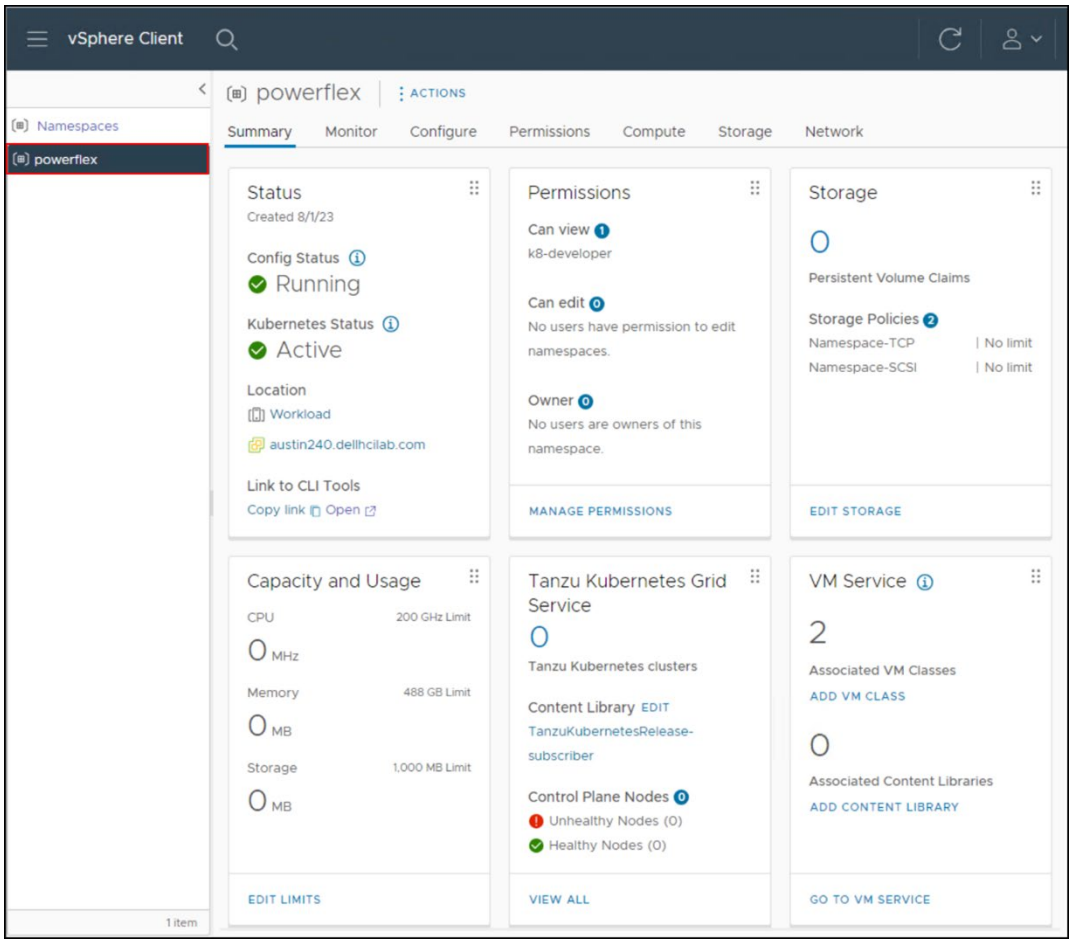

Figure 76. Completed PowerFlex namespace

CLI tools

The CLI tools enable a privileged user to create a Kubernetes cluster.

1. Now that the namespace exists, download the CLI Tools from the **Status** box as shown in Figure 76 above.
2. Click the **Open** hyperlink at the bottom of the Status box.

The Open hyperlink is pulled from the Control Plane Node Address of the Supervisor Cluster where the namespace was created, as shown in Figure 77.

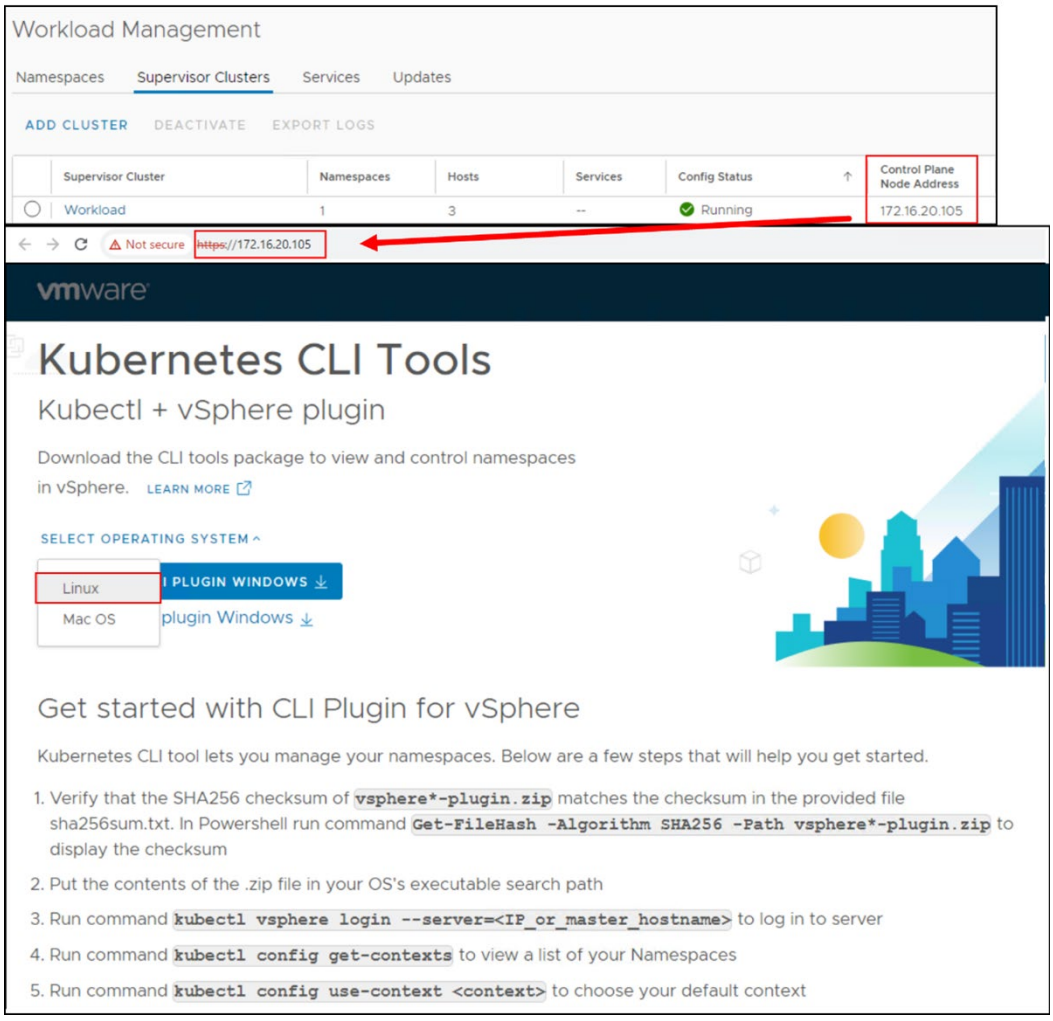

Figure 77. CLI Tools

The .zip file contains two binaries which can be copied into the proper operating system (for example, Linux) of a machine configured for the purpose. Utilizing these binaries, users who have been given permission to the namespace can access the environment.

Figure 77 above displays the steps for logging in to the namespace.

3. Log in to the Linux host where the user copied the binaries.
4. Following Step 3 as shown in Figure 77, login to the namespace. The password can be set before running the command by setting the variable **KUBECTL_VSPHERE_PASSWORD**.

```
kubectl-vsphere login --server 172.16.20.105 -u \
administrator@vsphere.local
```

Kubernetes cluster creation

1. Once connected, view the available contexts.
2. Set the current context to the namespace previously created.

```
kubectl config get-contexts
```

```
kubectl config use-context powerflex
```

```
root@austin202:~# kubectl-vsphere login --server 172.16.20.105 -u administrator@vsphere.local


Logged in successfully.

You have access to the following contexts:
   172.16.20.101
   172.16.20.105
   powerflex
   test

If the context you wish to use is not in this list, you may need to try
logging in again later, or contact your cluster administrator.

To change context, use `kubectl config use-context <workload name>`
root@austin202:~# kubectl config use-context powerflex
Switched to context "powerflex".
```

Figure 78. Logging in to a namespace and switching contexts

3. Check the available VM templates or VM classes and the available Tanzu images. This environment includes the **best effort small** and **guaranteed medium** images.

```
kubectl get virtualmachineclassbindings

kubectl get tkr
```

```
root@austin202:~# kubectl get virtualmachineclassbindings
NAME                AGE
best-effort-small   22d
guaranteed-medium   22d
root@austin202:~# kubectl get tkr
NAME                                VERSION                             READY   COMPATIBLE   CREATED   UPDATES AVAILABLE
v1.16.12---vmware.1-tkg.1.da7afe7   1.16.12+vmware.1-tkg.1.da7afe7      False   False        23d
v1.16.14---vmware.1-tkg.1.ada4837   1.16.14+vmware.1-tkg.1.ada4837      False   False        23d
v1.16.8---vmware.1-tkg.3.60d2ffd    1.16.8+vmware.1-tkg.3.60d2ffd       False   False        23d
v1.17.11---vmware.1-tkg.1.15f1e18   1.17.11+vmware.1-tkg.1.15f1e18      False   False        23d
v1.17.11---vmware.1-tkg.2.ad3d374   1.17.11+vmware.1-tkg.2.ad3d374      False   False        23d
v1.17.13---vmware.1-tkg.2.2c133ed   1.17.13+vmware.1-tkg.2.2c133ed      False   False        23d
v1.17.17---vmware.1-tkg.1.d44d45a   1.17.17+vmware.1-tkg.1.d44d45a      False   False        23d
v1.17.7---vmware.1-tkg.1.154236c    1.17.7+vmware.1-tkg.1.154236c       False   False        23d
v1.17.8---vmware.1-tkg.1.5417466    1.17.8+vmware.1-tkg.1.5417466       False   False        23d
v1.18.10---vmware.1-tkg.1.3a6cd48   1.18.10+vmware.1-tkg.1.3a6cd48      False   False        23d
v1.18.15---vmware.1-tkg.1.600e412   1.18.15+vmware.1-tkg.1.600e412      False   False        23d
v1.18.15---vmware.1-tkg.2.ebf6117   1.18.15+vmware.1-tkg.2.ebf6117      False   False        23d
v1.18.19---vmware.1-tkg.1.17af790   1.18.19+vmware.1-tkg.1.17af790      False   False        23d
v1.18.5---vmware.1-tkg.1.c40d30d    1.18.5+vmware.1-tkg.1.c40d30d       False   False        23d
v1.19.11---vmware.1-tkg.1.9d9b236   1.19.11+vmware.1-tkg.1.9d9b236      False   False        23d
v1.19.14---vmware.1-tkg.1.8753786   1.19.14+vmware.1-tkg.1.8753786      False   False        23d
v1.19.16---vmware.1-tkg.1.df910e2   1.19.16+vmware.1-tkg.1.df910e2      False   False        23d
v1.19.7---vmware.1-tkg.1.fc82c41    1.19.7+vmware.1-tkg.1.fc82c41       False   False        23d
v1.19.7---vmware.1-tkg.2.f52f85a    1.19.7+vmware.1-tkg.2.f52f85a       False   False        23d
v1.20.12---vmware.1-tkg.1.b9a42f3   1.20.12+vmware.1-tkg.1.b9a42f3      False   False        23d     [1.21.6+vmware.1-tkg.1.b3d708a]
v1.20.2---vmware.1-tkg.1.1d4f79a    1.20.2+vmware.1-tkg.1.1d4f79a       False   False        23d     [1.21.6+vmware.1-tkg.1.b3d708a]
v1.20.2---vmware.1-tkg.2.3e10706    1.20.2+vmware.1-tkg.2.3e10706       False   False        23d     [1.21.6+vmware.1-tkg.1.b3d708a]
v1.20.7---vmware.1-tkg.1.7fb9067    1.20.7+vmware.1-tkg.1.7fb9067       False   False        23d     [1.21.6+vmware.1-tkg.1.b3d708a]
v1.20.8---vmware.1-tkg.2            1.20.8+vmware.1-tkg.2               False   False        23d     [1.21.6+vmware.1-tkg.1]
v1.20.9---vmware.1-tkg.1.a4cee5b    1.20.9+vmware.1-tkg.1.a4cee5b       False   False        23d     [1.21.6+vmware.1-tkg.1.b3d708a]
v1.21.2---vmware.1-tkg.1.ee25d55    1.21.2+vmware.1-tkg.1.ee25d55       True    True         23d     [1.22.9+vmware.1-tkg.1.cc71bc8 1.21.6+vmware.1-tkg.1.b3d708a]
v1.21.6---vmware.1-tkg.1            1.21.6+vmware.1-tkg.1               True    True         23d     [1.22.9+vmware.1-tkg.1]
v1.21.6---vmware.1-tkg.1.b3d708a    1.21.6+vmware.1-tkg.1.b3d708a       True    True         23d     [1.22.9+vmware.1-tkg.1.cc71bc8]
v1.22.9---vmware.1-tkg.1            1.22.9+vmware.1-tkg.1               True    True         23d     [1.23.8+vmware.3-tkg.1.ubuntu]
v1.22.9---vmware.1-tkg.1.cc71bc8    1.22.9+vmware.1-tkg.1.cc71bc8       True    True         23d     [1.23.8+vmware.3-tkg.1]
v1.23.15---vmware.1                 1.23.15+vmware.1                    False   False        23d     [1.23.8+vmware.3-tkg.1]
v1.23.8---vmware.1-tkg.1            1.23.8+vmware.1-tkg.1               False   False        23d     [1.23.8+vmware.3-tkg.1.ubuntu]
v1.23.8---vmware.3-tkg.1            1.23.8+vmware.3-tkg.1               True    True         23d
v1.23.8---vmware.3-tkg.1.ubuntu     1.23.8+vmware.3-tkg.1.ubuntu        True    True         23d
v1.24.11---vmware.1-fips.1-tkg.1    1.24.11+vmware.1-fips.1-tkg.1       False   False        19d
v1.24.11---vmware.1-fips.1-tkg.1.ubuntu   1.24.11+vmware.1-fips.1-tkg.1.ubuntu   False   False   19d
v1.24.9---vmware.1                  1.24.9+vmware.1                     False   False        23d
v1.25.7---vmware.3-fips.1           1.25.7+vmware.3-fips.1              False   False        6d16h
```

Figure 79. List VM classes and Kubernetes builds

The user also requires the available storage classes. This information is listed in the UI. However, because of the previously mentioned restrictions around storage class naming, the classes that are listed in the UI may differ than what Kubernetes uses. For example, capital letters are not permitted for storage classes. Therefore, the two storage classes that are listed in the UI - **Namespace-SCSI** and **Namespace-TCP** - are renamed to **namespace-scsi** and **namespace-tcp** in Kubernetes.

4. Use the following command to pull the correct names for Kubernetes:

```
kubectl get sc
```

The user can use this information to create a Kubernetes cluster using TKGS in the **powerflex** namespace. There are two APIs available for creating the cluster: v1alpha1 API and v1alpha2 API.

VMware advises using v1alpha2 API since v1alpha1 API is deprecated. However, v1alpha2 API has some specific version requirements (for example, ESXi 7.0U3). If the environment does not meet them, use v1alpha1 API. The lab in this example meets the requirements, so the v1alpha2 API is used.

5. Create a YAML file with the necessary components to create the cluster.

Note: The following YAML is a basic cluster configuration. The most important adjustable parameters are bolded. Be sure that both the **vmClass** and **StorageClass** values are ones available for the chosen **namespace** (previously queried above).

```
apiVersion: run.tanzu.vmware.com/v1alpha2
kind: TanzuKubernetesCluster
metadata:
  name: pf-cluster
  namespace: powerflex
spec:
  topology:
    controlPlane:
      replicas: 1
      vmClass: guaranteed-medium
      storageClass: namespace-scsi
      tkr:
        reference:
          name: v1.23.8---vmware.3-tkg.1
    nodePools:
    - name: worker-nodes
      replicas: 3
      vmClass: guaranteed-medium
      storageClass: namespace-tcp
      tkr:
        reference:
          name: v1.23.8---vmware.3-tkg.1
```

6. Run the YAML file and then describe it, as shown in Figure 80.

```
root@austin202:~/tanzu# kubectl apply -f pf-cluster.v2.yaml
tanzukubernetescluster.run.tanzu.vmware.com/pf-cluster created
root@austin202:~/tanzu# kubectl describe tkc pf-cluster
Name:         pf-cluster
Namespace:    powerflex
Labels:       run.tanzu.vmware.com/tkr=v1.23.8---vmware.3-tkg.1
Annotations:  <none>
API Version:  run.tanzu.vmware.com/v1alpha2
Kind:         TanzuKubernetesCluster
Metadata:
  Creation Timestamp:  2023-08-03T15:31:34Z
  Finalizers:
    tanzukubernetescluster.run.tanzu.vmware.com
  Generation:  1
  Managed Fields:
    API Version:  run.tanzu.vmware.com/v1alpha2
    Fields Type:  FieldsV1
    fieldsV1:
      f:metadata:
        f:annotations:
          .:
          f:kubectl.kubernetes.io/last-applied-configuration:
      f:spec:
        .:
        f:topology:
          .:
          f:controlPlane:
            .:
            f:replicas:
            f:storageClass:
            f:tkr:
              .:
              f:reference:
                .:
                f:name:
            f:vmClass:
          f:nodePools:
    Manager:      kubectl-client-side-apply
    Operation:    Update
    Time:         2023-08-03T15:31:34Z
    API Version:  run.tanzu.vmware.com/v1alpha2
```

Figure 80. Creating a Tanzu Kubernetes cluster

The user can monitor the creation of the cluster from the namespace within vCenter.

7. Under the **Compute** tab, view the **VMware Resources – Tanzu Kubernetes cluster** and **Virtual Machines** as shown in Figure 81.

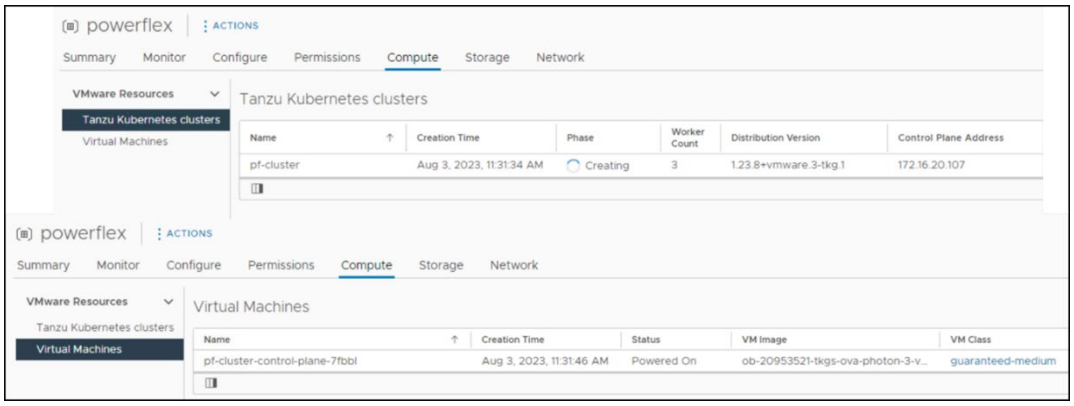

Figure 81. Monitoring cluster creation in the namespace

When cluster creation is complete, the **Phase** shows **Running**, as shown in Figure 82.

Note: As shown in the red box around the Control Plane Address, the load balancer provided the workload IP address.

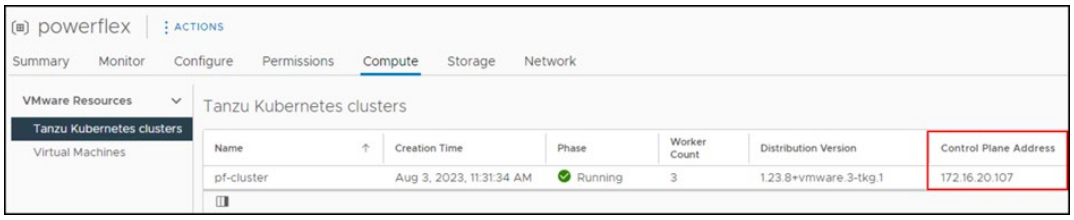

Figure 82. Running Kubernetes cluster

8. With the new cluster available, login to the namespace and cluster using the following command:

```
kubectl-vsphere login --server 172.16.20.105 -u \
administrator@vsphere.local --tanzu-kubernetes-cluster-namespace \
powerflex --tanzu-kubernetes-cluster-name pf-cluster
```

The output is shown in Figure 83.

```
root@austin202:~# kubectl-vsphere login --server 172.16.20.105 -u administrator@vsphere.local --tanzu-kubernetes-cluster-namespace powerflex
--tanzu-kubernetes-cluster-name pf-cluster


Logged in successfully.

You have access to the following contexts:
   172.16.20.101
   172.16.20.105
   pf-cluster
   powerflex
   test

If the context you wish to use is not in this list, you may need to try
logging in again later, or contact your cluster administrator.

To change context, use `kubectl config use-context <workload name>`
root@austin202:~# kubectl config get-contexts
CURRENT   NAME            CLUSTER         AUTHINFO                                        NAMESPACE
          172.16.20.101   172.16.20.101   wcp:172.16.20.101:administrator@vsphere.local
          172.16.20.105   172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local
*         pf-cluster      172.16.20.107   wcp:172.16.20.107:administrator@vsphere.local
          powerflex       172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local   powerflex
          test            172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local   test
root@austin202:~#
```

Figure 83. Log in to the new cluster

At this point, applications can be deployed on the new cluster. For completeness, an example is available in Appendix A, Application deployment.

Chapter 7 Conclusion

This chapter presents the following topics:

Summary

VMware vSphere with Tanzu provides customers the flexibility to combine a traditional VM infrastructure with a robust container implementation based on the industry standard Kubernetes. PowerFlex storage enables the Tanzu environment to become highly resilient and scalable, able to handle the most demanding workloads whether on virtual machines or Kubernetes.

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell Technologies Solutions team by email.

Authors: Drew Tonnesen, Dell Technologies PowerFlex Engineering

Contributors: Dell Technologies Solution Information Development & Design team

Chapter 8 References

This chapter presents the following topics:

Dell Technologies documentation

The following Dell Technologies documentation provides additional and relevant information. Access to these documents depends on your login credentials. If you do not have access to a document, contact your Dell Technologies representative.

- Dell PowerFlex documentation

VMware documentation

The following VMware documentation provides additional and relevant information:

- VMware Tanzu documentation

Kubernetes documentation

The following Microsoft documentation provides additional and relevant information:

- Kubernetes documentation

Appendix A Application deployment

This appendix presents the following topics:

Introduction

Once you deploy a Tanzu Kubernetes cluster, developers with access can install applications on it. This appendix offers an example that Kubernetes provides for you to duplicate to validate the environment.

This example deploys a WordPress website, backed by a MySQL database that is taken from the following Kubernetes documentation example:

Deploying WordPress and MySQL with Persistent Volumes

The WordPress deployment takes advantage of the ALB to receive a virtual IP (VIP) by which users can access the application. The same Kubernetes cluster that is created in the guide is used, **pf-cluster**.

Kustomize

Kustomize is a stand-alone tool that is part of Kubernetes and can be used to customize objects. It lets users manage objects like secrets and deployments in a single file. It is used here.

Prerequisites

There are three parts to the deployment:

- A secret generator
- A MySQL deployment
- A WordPress deployment

Secret generator

The secret generator is placed directly in the `kustomization` file. The bolded value should be adjusted for personal preference:

```
secretGenerator:

- name: mysql-pass

  literals:

  - password=oracle
```

The other two parts are referenced in the `kustomization` file as resources, but the YAML files must exist for them.

MySQL YAML

The bolded items require customization according to the user environment. Any referenced objects (like storage classes) must exist in the chosen namespace.

`mysql-deployment.yaml`

```
apiVersion: v1

kind: Service
```

```
metadata:
  name: wordpress-mysql
  labels:
    app: wordpress
spec:
  ports:
    - port: 3306
  selector:
    app: wordpress
    tier: mysql
  clusterIP: None
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
  labels:
    app: wordpress
spec:
  storageClassName: namespace-tcp
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
```

```
    name: wordpress-mysql
    labels:
      app: wordpress
  spec:
    selector:
      matchLabels:
        app: wordpress
        tier: mysql
    strategy:
      type: Recreate
    template:
      metadata:
        labels:
          app: wordpress
          tier: mysql
      spec:
        containers:
        - image: mysql:8.0
          name: mysql
          env:
          - name: MYSQL_ROOT_PASSWORD
            valueFrom:
              secretKeyRef:
                name: mysql-pass
                key: password
          - name: MYSQL_DATABASE
            value: wordpress
          - name: MYSQL_USER
            value: wordpress
```

```
        - name: MYSQL_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim
```

WordPress YAML

The bolded items require customization according to the user environment. Any referenced objects (like storage classes) must exist in the chosen namespace.

wordpress-deployment.yaml

```
apiVersion: v1
kind: Service
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  ports:
    - port: 80
  selector:
```

```
    app: wordpress
    tier: frontend
  type: LoadBalancer
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: wp-pv-claim
  labels:
    app: wordpress
spec:
  storageClassName: namespace-tcp
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 20Gi
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wordpress
  labels:
    app: wordpress
spec:
  selector:
    matchLabels:
      app: wordpress
      tier: frontend
```

```
strategy:
  type: Recreate
template:
  metadata:
    labels:
      app: wordpress
      tier: frontend
  spec:
    containers:
    - image: wordpress:6.2.1-apache
      name: wordpress
      env:
      - name: WORDPRESS_DB_HOST
        value: wordpress-mysql
      - name: WORDPRESS_DB_PASSWORD
        valueFrom:
          secretKeyRef:
            name: mysql-pass
            key: password
      - name: WORDPRESS_DB_USER
        value: wordpress
      ports:
      - containerPort: 80
        name: wordpress
      volumeMounts:
      - name: wordpress-persistent-storage
        mountPath: /var/www/html
    volumes:
    - name: wordpress-persistent-storage
```

```
        persistentVolumeClaim:

          claimName: wp-pv-claim
```

Now create the kustomization file to create the secret generator then execute the two YAMLs.

```
secretGenerator:

- name: mysql-pass

  literals:

  - password=oracle

resources:

  - mysql-deployment.yaml

  - wordpress-deployment.yaml
```

Procedure

1. Log in to the **pf-cluster**, apply the file, and view the various objects as shown in Figure 84.

```
root@austin202:~# kubectl-vsphere login --server 172.16.20.105 -u administrator@vsphere.local --tanzu-kubernetes-cluster-namespace powerflex
--tanzu-kubernetes-cluster-name pf-cluster


Logged in successfully.

You have access to the following contexts:
   172.16.20.101
   172.16.20.105
   pf-cluster
   powerflex
   test

If the context you wish to use is not in this list, you may need to try
logging in again later, or contact your cluster administrator.

To change context, use `kubectl config use-context <workload name>`
root@austin202:~# kubectl config get-contexts
CURRENT   NAME            CLUSTER         AUTHINFO                                        NAMESPACE
          172.16.20.101   172.16.20.101   wcp:172.16.20.101:administrator@vsphere.local
          172.16.20.105   172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local
*         pf-cluster      172.16.20.107   wcp:172.16.20.107:administrator@vsphere.local
          powerflex       172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local   powerflex
          test            172.16.20.105   wcp:172.16.20.105:administrator@vsphere.local   test
root@austin202:~# cd tanzu
root@austin202:~/tanzu# kubectl apply -k ./
secret/mysql-pass-t4tkcb9cd2 created
service/wordpress created
service/wordpress-mysql created
persistentvolumeclaim/mysql-pv-claim created
persistentvolumeclaim/wp-pv-claim created
deployment.apps/wordpress created
deployment.apps/wordpress-mysql created
root@austin202:~/tanzu# kubectl get secrets
NAME                  TYPE                                  DATA   AGE
default-token-vsqsb   kubernetes.io/service-account-token   3      137m
mysql-pass-t4tkcb9cd2   Opaque                              1      14s
root@austin202:~/tanzu# kubectl get pvc
NAME             STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS    AGE
mysql-pv-claim   Bound    pvc-ab4a8bfc-e7e3-421b-bb4e-c768d7427541   20Gi       RWO            namespace-tcp   35s
wp-pv-claim      Bound    pvc-a9a86a92-4e8c-4520-8a08-c6e476642049   20Gi       RWO            namespace-tcp   35s
```

Figure 84. Kustomize application

2. View the resources of the namespace in vCenter. Note the two new persistent volume claims, as shown in Figure 85.

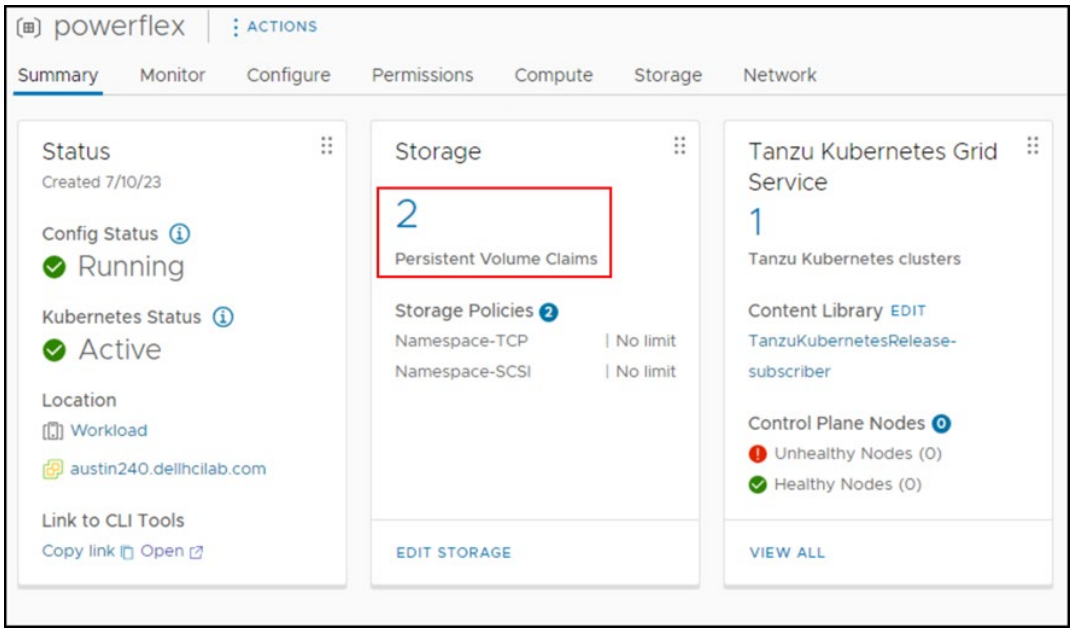

Figure 85. Persistent Volume Claims

3. Click the hyperlink to get details about the volumes as seen in Figure 86.

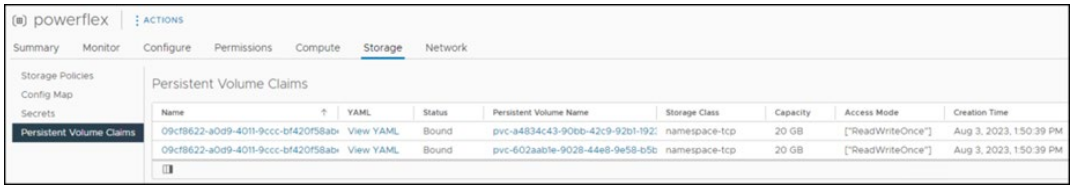

Figure 86. PVC details

4. Retrieve the ALB VIP that is associated with the WordPress application in Figure 87. This VIP is for the workload network and is accessible externally.

```
root@austin202:~# kubectl get services wordpress
NAME        TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)        AGE
wordpress   LoadBalancer   10.108.217.162  172.16.20.109   80:30141/TCP   31m
```

Figure 87. ALB VIP of WordPress

This IP address and port can be used to access the application through a browser. The **EXTERNAL-IP** uses port 80 as seen in Figure 88.

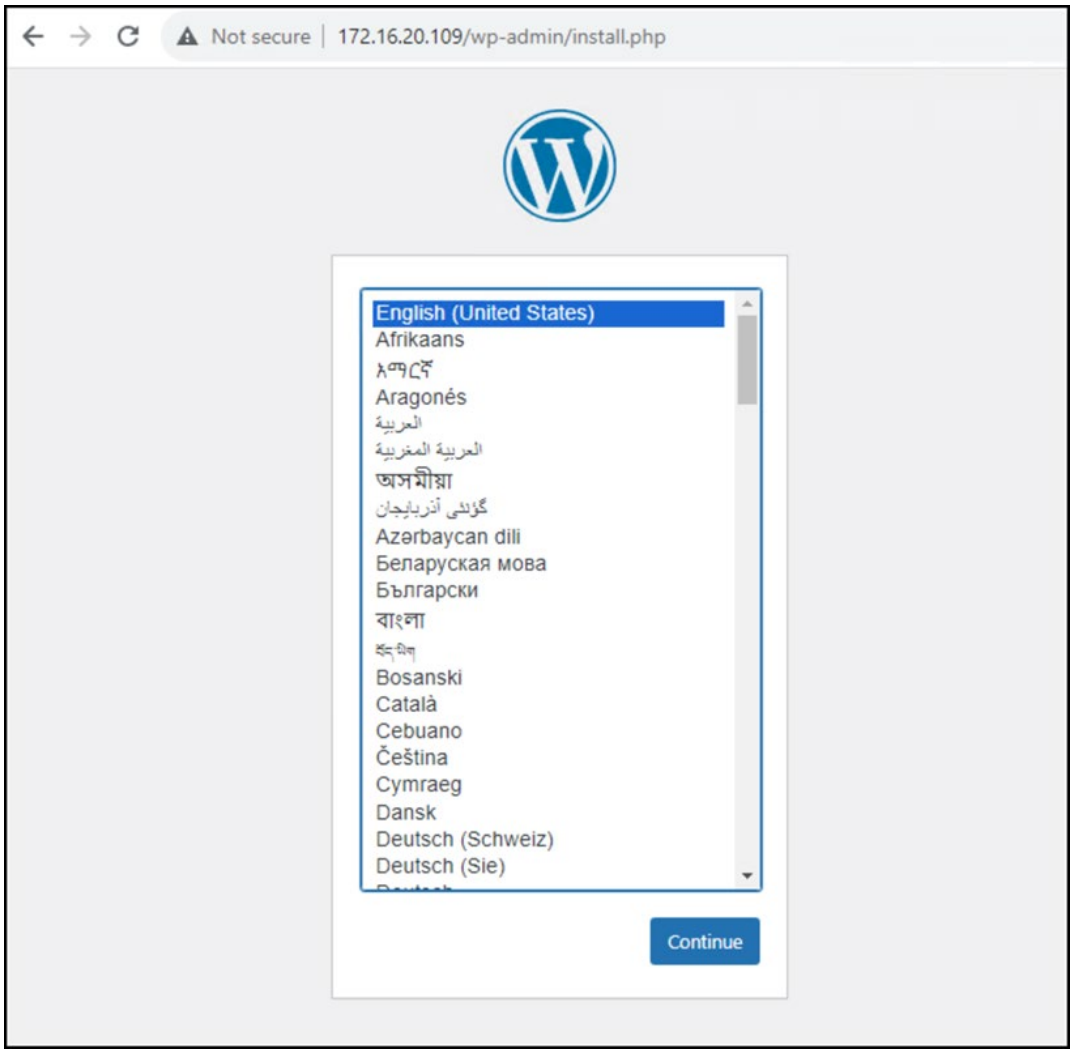

Figure 88. WordPress wizard

5. Click **Continue** to go through and complete the wizard.

The install is complete as shown in Figure 89.

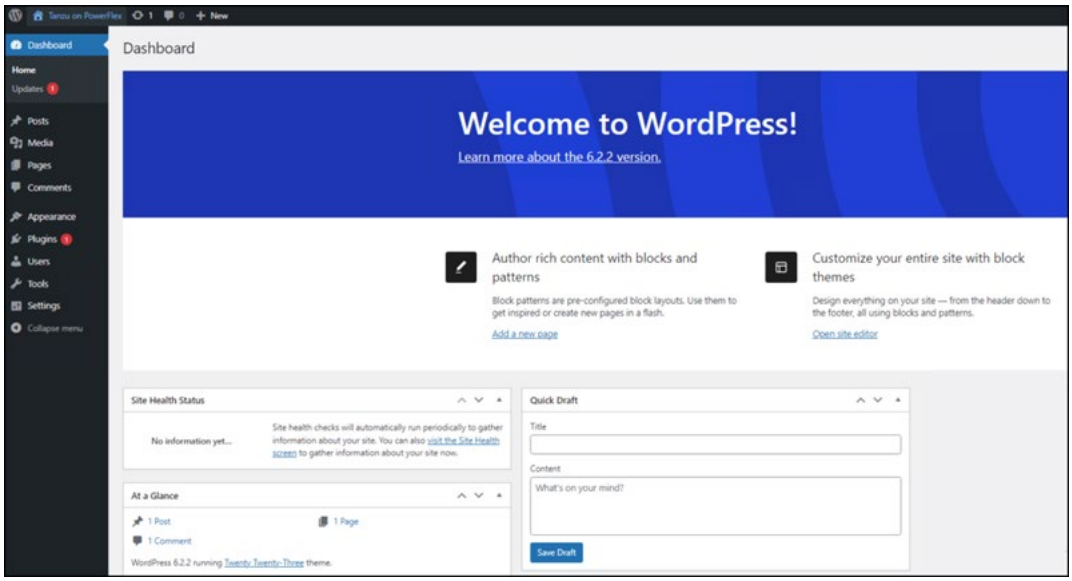

Figure 89. WordPress completed application