MinIO Object Storage on VMware Cloud Foundation with Tanzu

High performance, cloud native object storage with the VMware vSAN Data Persistence platform

vmware

MINIO

Table of contents

Executive Summary

Business Case

The enterprise is undergoing an exceptional period of change. Driven by the relentless growth of data, the emergence of Kubernetes and the demands of the hybrid cloud, both IT and DevOps teams face challenges that are both massive and, inherently cross disciplinary.

VMware Cloud Foundation™ with Tanzu™ accelerates Kubernetes infrastructure provisioning with full stack consisting of compute, storage, networking, and management. Through automatic and reliable deployment of multiple workload domains, it increases admin productivity while reducing overall TCO to deliver a faster path to a hybrid cloud.

The VMware vSAN™ Data Persistence platform (DPp) is an as-a-service framework to run these modern stateful services with vSphere integration to achieve lower TCO, higher performance, and simplified operational management. The framework leverages Kubernetes operators of VMware partners and VMware vSphere® Pod Service to offer this value. The deep integrations make the service appear native to vSphere while still offering simple cloud like self-service consumption to developers.

MinIO and VMware have partnered to design a framework that brings cloud native object storage directly into the heart of VMware vSphere through the development of the Data Persistence platform. Through the DPp, MinIO gains access to the underlying storage via VMware vSAN Direct Configuration™, a technology that provides select stateful services with direct access to the underlying disks. This enables IT admins to provision peta-scale infrastructure with a few clicks in the VMware vCenter® interface. Through this direct datapath, MinIO can drive the maximum read and write performance available with VMware vSAN Direct and vSAN-SNA (vSAN support for Shared Nothing Architecture). Furthermore, because MinIO can operate at the supervisor cluster level of vSphere with Tanzu, enterprises can create extremely large-scale object storage clusters with minimal friction. MinIO Object Storage is fully integrated with vSphere Day 2 workflows. For IT admins, managing MinIO Object Storage is as easy as managing VMs. As a result, IT admins will be able to provision, expand, and monitor high-performance, multi-tenant object storage for a range of applications and use cases and across internal and external stakeholders – directly from the vCenter interface, thereby accelerating the adoption of Kubernetes and digital transformation.

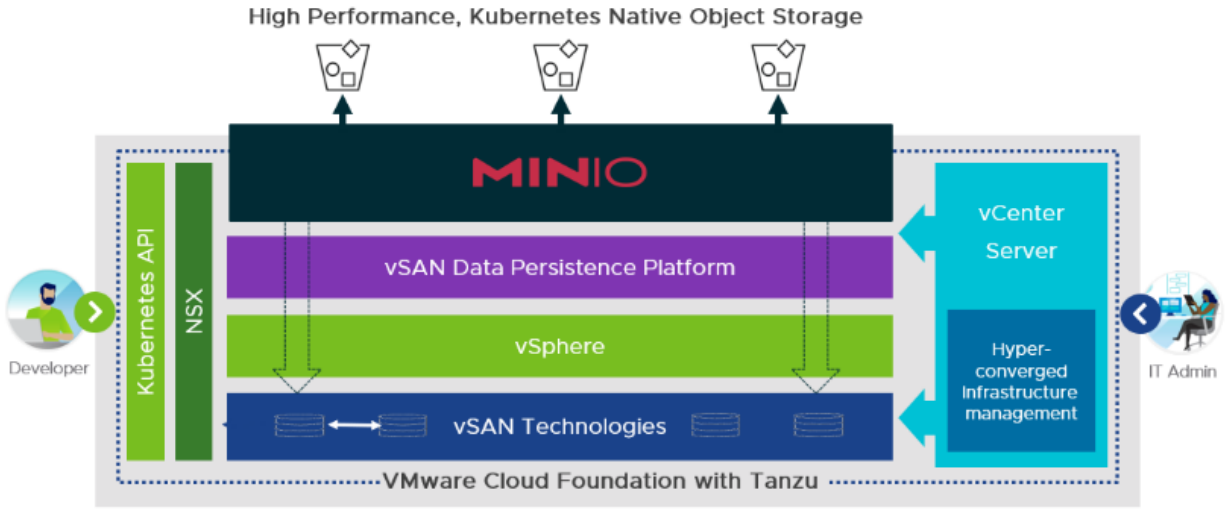


Figure 1. MinIO on VMware Cloud Foundation with Tanzu

In this solution, we provide deployment procedures, design and sizing guidance, best practices for enterprise infrastructure admins and application owners to run MinIO object storage on the Cloud Foundation platform.

Business Values

Here are top 5 benefits for deploying MinIO object storage on VMware Cloud Foundation with Tanzu using vSAN Data Persistence platform：

- Multi-tenant Object Storage from vCenter

  IT admins can now deploy high-performance, multi-tenant object storage through VMware vCenter with minimal knowledge or experience with the underlying Kubernetes infrastructure.

- Hybrid and Multi Cloud

  MinIO helps advance VMware's goal of letting enterprises build, run, manage, connect, and protect any app on any cloud or across clouds with complete consistency of experience. Specifically, MinIO fully supports both cloud native infrastructure and existing legacy storage systems using a common interface. Both cloud native and traditional enterprise applications can co-exist using MinIO to bridge the experience between environments.

- A Portfolio of Cloud Native Applications

  MinIO was designed to be cloud native and has thousands of existing integrations with other cloud native or S3-compatible applications, all of which become accessible to DPp users on Day One without modification.

- Performance

  MinIO has built its reputation in the private cloud as the world's fastest object store. MinIO uses the vSAN Direct Configuration architecture to gain direct access to underlying drives in JBOD/F mode, while retaining ownership of key storage functions like Erasure Coding, Bitrot Protection, and Encryption Key Management. While VMware focuses on the physical-to-virtual block layer management tasks, MinIO handles providing industry-leading object storage performance and efficiency.

- Manageability

  One of the key advantages of Kubernetes is the ability to manage large scale infrastructure with fewer resources. This promise can only be achieved if the storage infrastructure delivers simplicity and manageability. Simplicity is quite hard - it requires discipline and commitment, but the payoff is scalability - across the technology and business vectors. Minio's API-based, infrastructure-as-code architecture delivers elasticity while acting as a management force-multiplier through a relentless focus on automation.

Audience

This solution is intended for IT admins and storage experts who are involved in planning, designing, and deploying Object Storage resources on VMware Cloud Foundation through VMware vCenter. This document assumes that the reader is familiar with the concepts and operations of MinIO and VMware Cloud Foundation-related components.

Technology Overview

Solution technology components are listed below:

- VMware Cloud Foundation
  - VMware Cloud Foundation with Tanzu
  - VMware vSphere
  - VMware vSAN
  - VMware vSAN Data Persistence platform
  - VMware NSX Data Center
- MinIO Object Storage

VMware Cloud Foundation with Tanzu

VMware Cloud Foundation with Tanzu is the best way to deploy Kubernetes at scale. VMware Cloud Foundation is a ubiquitous hybrid cloud platform for both traditional enterprise apps and modern apps, providing a complete set of secure software-defined services for compute, storage, network security, Kubernetes management, and cloud management. VMware Cloud Foundation with Tanzu automates full-stack deployment and operation of Kubernetes clusters through integration with VMware Tanzu Kubernetes Grid. This helps eliminate manual steps for configuring hosts, creating logical relationships, managing hypervisors for faster deployment of applications at scale. The most exciting feature added to the VMware Cloud Foundation architecture is the integration of Kubernetes directly into the vSphere hypervisor, which delivers an entirely new set of VMware Cloud Foundation Services, a new Kubernetes and RESTful API surface that empowers developers to have self-service access to Kubernetes clusters, vSphere Pods, virtual machines, persistent volumes, stateful services, networking resources, etc. The result is an agile, reliable and efficient hybrid cloud platform that bridges the gap between app developers and IT admins.

VMware vSphere

VMware vSphere is the next-generation infrastructure for next-generation applications, which provides a powerful, flexible, and secure foundation for business agility that accelerates the digital transformation to cloud computing and promotes success in the digital economy. VMware vSphere embeds containers and Kubernetes into vSphere, unifying them with virtual machines as first-class citizens. This enables all vSphere admins to become Kubernetes admins and easily deliver new services to their developers. VMware vSphere addresses key challenges faced by the IT admins in areas of lifecycle management, security, and performance and resiliency needed by business-critical applications, AI/ML applications and latency sensitive applications. With VMware vSphere, customers can run, manage, connect, and secure both traditional and cloud native applications in a common operating environment, across clouds and devices.

VMware vSAN

VMware vSAN is the industry-leading software powering VMware's software defined storage and HCI solution. vSAN helps customers evolve their data center with reduced risk, control IT costs, and scale to tomorrow's business needs. vSAN, native to the market-leading hypervisor, delivers flash-optimized, secure storage for all of your critical vSphere workloads, and is built on industry-standard x86 servers and components that help lower TCO in comparison to traditional storage.

vSAN simplifies Day 1 and Day 2 operations, and customers can quickly deploy and extend cloud infrastructure and minimize maintenance disruptions. Together with MinIO Object Storage, vSAN modernizes HCI by providing admins a unified storage control plane for all block, file and object protocols and provides significant enhancements that make it a great solution for VMs as well cloud native applications. vSAN helps reduce the complexity of monitoring and maintaining infrastructure and enables admins to rapidly provision storage for Kubernetes-orchestrated cloud native applications.

VMware vSAN Data Persistence platform

The vSAN Data Persistence platform provides an as-a-service framework for VMware partners that offer modern stateful services to integrate with the underlying virtual infrastructure, allowing you to run stateful services with high velocity scaling, simplified IT operations, and optimized TCO. You can deploy a stateful service alongside traditional applications on a regular vSAN cluster with vSAN-SNA (vSAN for Shared Nothing Architecture) policy, or deploy it on a dedicated vSAN cluster with vSAN Direct configuration, a technology enabling direct access to the underlying direct-attached hardware which can be optimized for the application needs. Both options benefit from optimal storage efficiency for stateful services by leveraging service-level replication, as well as unified management of services in vCenter.

The platform offers a way for the IT admin to enable, manage and monitor all aspects of the stateful service from vSphere Interfaces (API/UI) while the developers get public cloud-like simple self-service consumption experience.

The regular vSAN cluster with vSAN-SNA policy or a dedicated vSAN cluster with vSAN Direct Configuration makes it easy for cloud native services such as MinIO integrated into the vSAN Data Persistence platform to co-locate its compute and storage on the same physical ESXi host. This host-local placement then allows us to do replication only at the service layer and not at the storage layer.

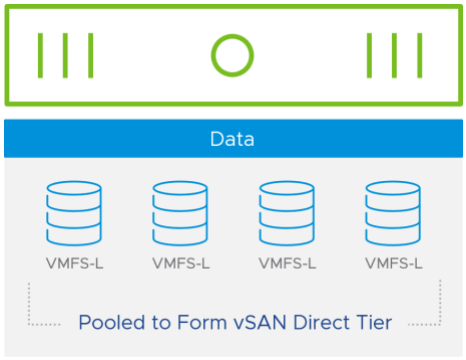
- vSAN with SNA Storage Policy

vSAN with SNA storage policy allows for using a distributed replicated vSAN datastore with the vSAN host-local SNA policy. The technology makes it easy for the stateful service to co-locate its compute instance and a storage object on the same physical ESXi host. The compute instance such as a pod has to come up first on one of the nodes in the vSAN cluster and then the vSAN object created with vSAN-SNA policy (vsan-sna storage class) will automatically have all of its data placed on the same node where the pod is running.

Although vSAN with SNA storage policy can place data local to the compute, we still leverage a distributed vSAN data path between the application and the raw physical disk. Also, the application can only specify affinity at the node granularity and cannot use different disks attached to the same node as independent fault domains.

- vSAN Direct Configuration

vSAN Direct Configuration provides optimal host local storage to shared nothing cloud native services, it creates independent datastores on every disk attached to a physical host and makes it available as a placement choice to modern applications. vSAN Direct Configuration extends the simplicity of HCI management to host local VMFS disks. It manages and monitors VMFS-L disks and provides insights into health, performance, and capacity of these disks. The VMFS datastores that vSAN Direct Configuration manages are exposed as storage pools in Kubernetes. vSAN Direct Configuration allows modern applications direct access to the disks while giving the ease of management to the VI admin, these disks are consumed just like vSAN with minimal overhead.



VMware NSX Data Center

VMware NSX® Data Center is the network virtualization and security platform that enables the virtual cloud network, a software-defined approach to networking that extends across data centers, clouds, and application frameworks. With NSX Data Center, networking and security are brought closer to the application wherever it's running, from virtual machines to containers to bare metal. Like the operational model of VMs, networks can be provisioned and managed independent of underlying hardware. NSX Data Center reproduces the entire network model in software, enabling any network topology—from simple to complex multitier networks—to be created and provisioned in seconds. Users can create multiple virtual networks with diverse requirements, leveraging a combination of the services offered via NSX or from a broad ecosystem of third-party integrations ranging from next-generation firewalls to performance management solutions to build inherently more agile and secure environments. These services can then be extended to a variety of endpoints within and across clouds.

MinIO

MinIO is a Kubernetes-native distributed object storage server designed to provide high performance on peta-scale infrastructure. MinIO has built its reputation in the private cloud as the world's fastest object store and is capable of utilizing the maximum possible throughput of vSAN Direct or vSAN SNA drives. Admins can leverage VMWare Tanzu Kubernetes orchestration to provision multi-tenant MinIO object storage to internal or external stakeholders entirely through the vCenter interface. Developers can rely on MinIO's best-in-class S3-compatible API when migrating applications from single-cloud or legacy infrastructures to MinIO-backed hybrid cloud object storage.

MinIO offers out-of-the-box enterprise-grade data security, availability, and redundancy features with minimal extra configuration. MinIO Erasure Coding provides configurable availability and redundancy settings, where the highest redundancy levels allow a MinIO tenant to continue serving read and write operations despite the loss of up to half the drives in the cluster. MinIO TLS and optional Server-Side Encryption protects data on disk and over the wire, while Write-Once Read-Many (WORM) object locking provides compliance and retention controls.

See *MinIO* for more information.

Solution Configuration

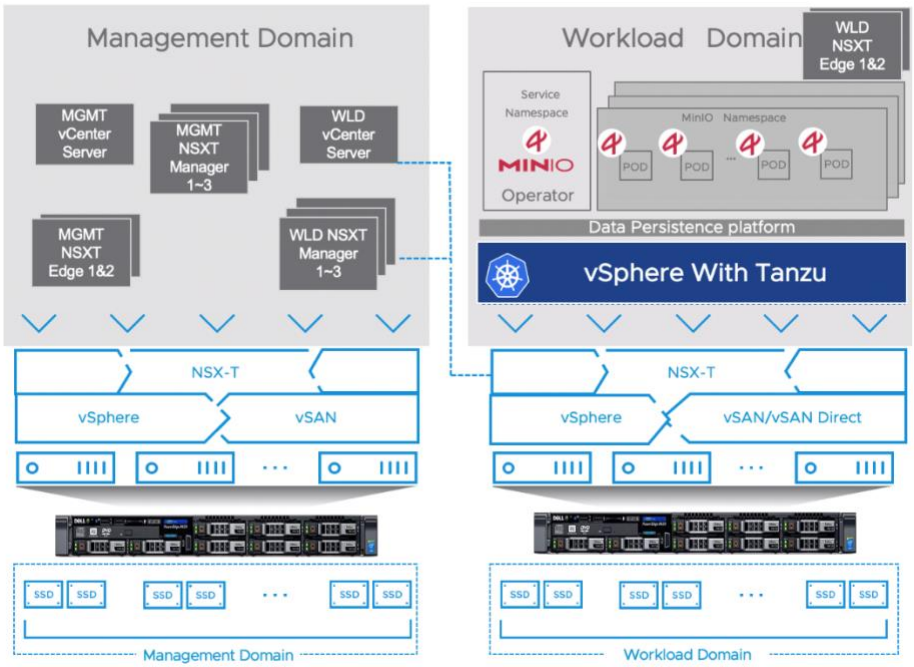
This section introduces the resources and configurations:

- Architecture diagram
- Hardware resources
- Software resources
- Network configuration

Architecture Diagram

The following sections describe how the MinIO Object Storage components are configured.

MinIO is typically deployed on a workload domain[1] with a recommended N+1 minimum of 4-node cluster. The following example management domain consists of a 4-node cluster shared across all workload domains.



1. In VMware Cloud Foundation, a workload domain is a policy-based resource construct with specific availability and performance attributes. It combines compute (vSphere), storage (vSAN), networking (NSX), and cloud management (vRealize Suite) into a single consumable entity. Workload domains greatly speed up the instantiation of Kubernetes clusters, deploying the underlying infrastructure in an automated fashion. Workload domains also allow IT admins and developers to securely sandbox and allocate the right infrastructure for containers alongside VMs.

Figure 2. Architectural Diagram

Hardware Resources

MinIO is hardware agnostic and runs on a variety of hardware architectures ranging from ARM-based embedded systems to high-end x64 and POWER9 servers. The following table provides a general description of a server provisioned for use with MinIO. For specific hardware recommendations, visit the *MinIO Reference Hardware* page.

Table 1. Hardware Configuration

PROPERTY	SPECIFICATION
CPU	MinIO is not CPU bound and is performant with any modern server-grade CPU. MinIO can take advantage of the AVX-512 SIMD acceleration offered by Intel Skylake processors for increased performance of certain operations.
RAM	MinIO is not memory bound and is performant with any modern speed memory. Larger datasets benefit from more available memory.
Network adapter	Minimum 10GbE NIC (Supports ~1GBps of aggregated storage throughput).
Storage adapter	Dedicated SAS/SATA Storage Controllers with JBOD support and RAID disabled.
Disks	Dedicated SAS/SATA drives. MinIO can drive the full throughput provided by vSAN Direct and vSAN SNA.

MinIO is typically throughput bound rather than compute bound, where the network and storage layers are most likely to dictate the total object storage throughput. Specifically, throughput constraints tend to be in the following order:

Network -> Storage -> Storage Controller/PCIe Bus

For example, a 10GbE network infrastructure can support ~1GBps of aggregated storage throughput. While the aggregated storage, storage controller, or PCIe bus may support higher potential throughput, MinIO is constrained by the network infrastructure. For more complete guidance on sizing hardware to support MinIO, see the Sizing Guidance section.

Software Resources

Table 2 shows the software resources used in this solution.

Table 2. Software Resources

SOFTWARE	VERSION	PURPOSE
VMware Cloud Foundation	4.2	VMware Cloud Foundation provides integrated cloud infrastructure (compute, storage, networking, and security) and cloud management services to run both modern, cloud native and traditional workloads. See *VMware Cloud Foundation* for details.
MinIO Plugin	1.05	The MinIO Plugin supports deploying multi-tenant object storage on VMware Tanzu.

MinIO Server	minio/minio:RELEASE.*2020-10-18T21-54-12Z*	MinIO High-Performance Object Storage with AWS S3-compatibility

Network Configuration

Table 3. Virtual Distributed Switch Teaming Policy

Port Group	Teaming Policy	VMNIC0	VMNIC1
Management network	Route based on Physical NIC load	Active	Active
VM network	Route based on Physical NIC load	Active	Active
vSphere vMotion	Route based on Physical NIC load	Active	Active
vSAN	Route based on Physical NIC load	Active	Active
VXLAN VTEP	Route based on the originating virtual port	Active	Active

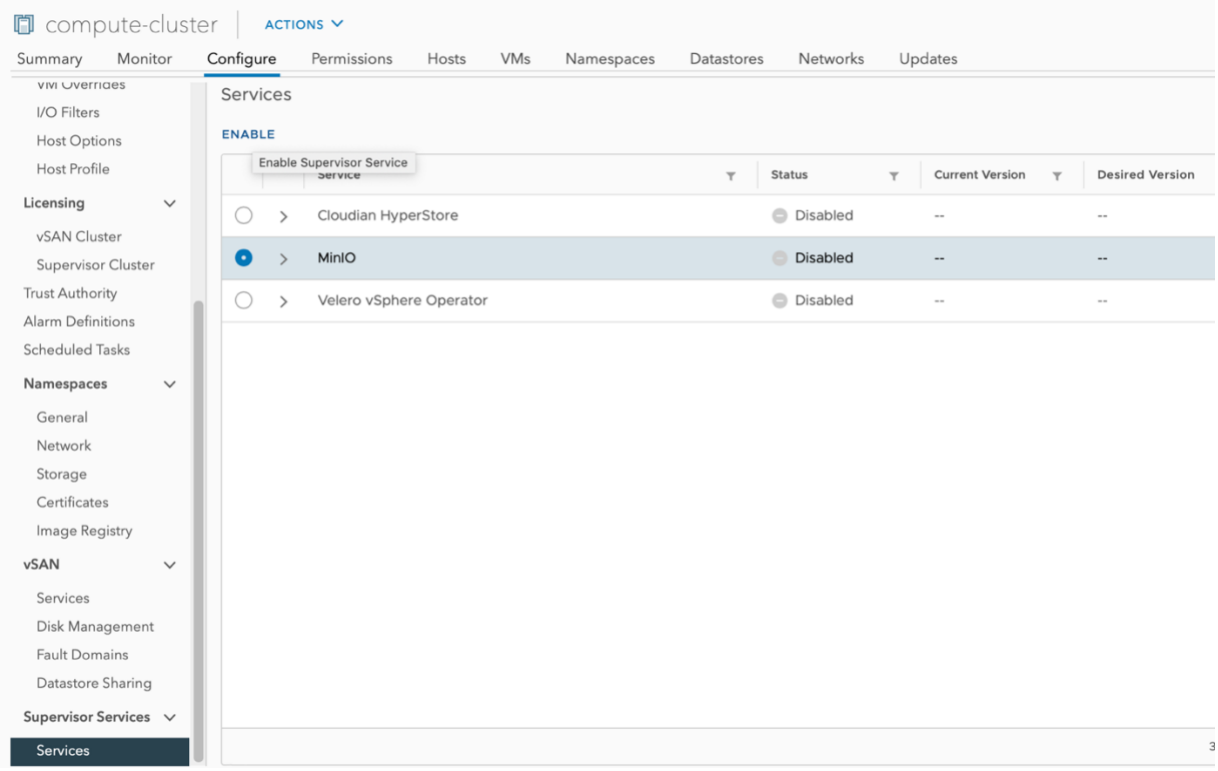
The Supervisor Cluster that is configured with VMware NSX-T Data Center, uses the software-based networks of the solution as well as an NSX Edge load balancer to provide connectivity to external services and DevOps users. The MinIO operator pods are placed in a namespace, MinIO tenant pods are placed in namespaces, for information, see Namespaces Networking.

User Guide

Overview

With the vSAN Data Persistence platform, service operators can be deployed with a single click by the VI admin straight from the vSphere UI. In brief, the platform operator brings up an operator of the service that is enabled from vSphere UI, and the service can register a vCenter UI plugin, which offers a seamless user experience to manage these services right within vSphere.

The platform operator creates optimized storage policies for this service, which can be used to deploy service instances. Once the service operator is up, developers can provision and scale instances of the service either through UI or Kubernetes APIs in a self-service manner.



Since modern stateful services usually provide their own replication, the vSAN Data Persistence platform provides two new vSAN data paths (vSAN with SNA storage policy and vSAN Direct) specially designed for cloud native stateful services to co-locate its compute and storage on the same physical ESXi host. This host-local placement then allows us to do replication only at the service layer and not at the storage layer. Using vSAN with SNA storage policy is prudent when the cloud native stateful application shares the physical infrastructure with other regular VM or k8s workload. Since each workload can define its own storage policy and can get the best of both worlds from a single cluster. We recommend customers to use vSAN Direct deployment if they are creating a dedicated hardware cluster for such shared nothing cloud native services.

For vSAN direct configuration, see *Set Up vSAN Direct for vSphere with Tanzu*. The following steps document creating a MinIO tenant on VMware Cloud Foundation 4.2. The procedure assumes that the vSphere infrastructure includes **at minimum** the following resources:

- ESXi hosts with locally-attached storage (for example: SATA/SAS HDD/SSD drives).
- vSAN Storage Policy that allocates local storage drive using vSAN Direct or vSAN SNA.
- Kubernetes Namespace on which the MinIO tenant deploys. MinIO supports at most one MinIO tenant per namespace. The namespace must be configured with access to the vSAN Direct or vSAN-SNA storage policy.

Key Results

This procedure is a showcase of VMware Cloud Foundation with Tanzu for operating and managing high performance, cloud native MinIO object storage using the vSAN Data Persistence platform in a fully integrated SDDC environment. Key results can be summarized as follows:

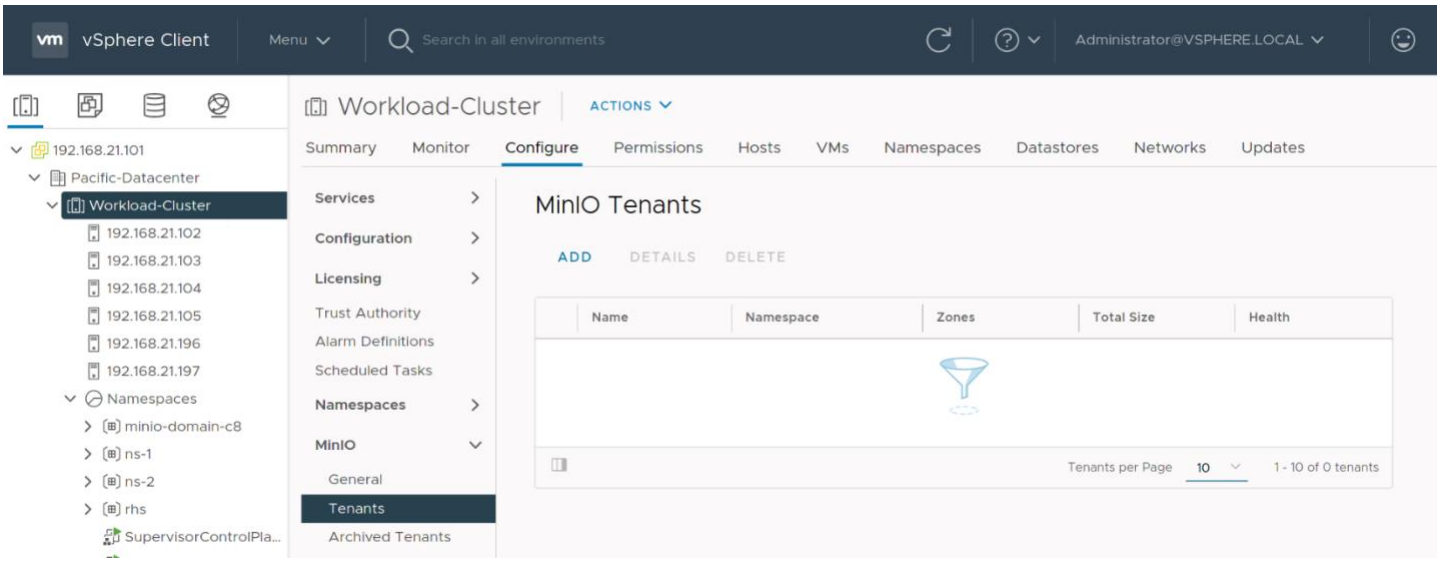
- Deploying Tenant ---Quick guide through creating a MinIO tenant-service health and capacity monitoring.
- Resilience Tests---Proves solution resilience to guarantee MinIO service continuity and stability in failure scenarios such as disk and host failures.

- Best Practices---Provides best practices to deploy the infrastructure and sizing guidelines of CPU/memory/storage/network bandwidth planning to target a given usable capacity and throughput.
- Top Use cases---Addresses a broad range of use cases from backup/snapshots/archival, AI and ML, web/mobile cloud native application development, analytics to artifact stores, given MinIO's performance and scalability attributes, results in superior return on investment.

Create a New MinIO Tenant

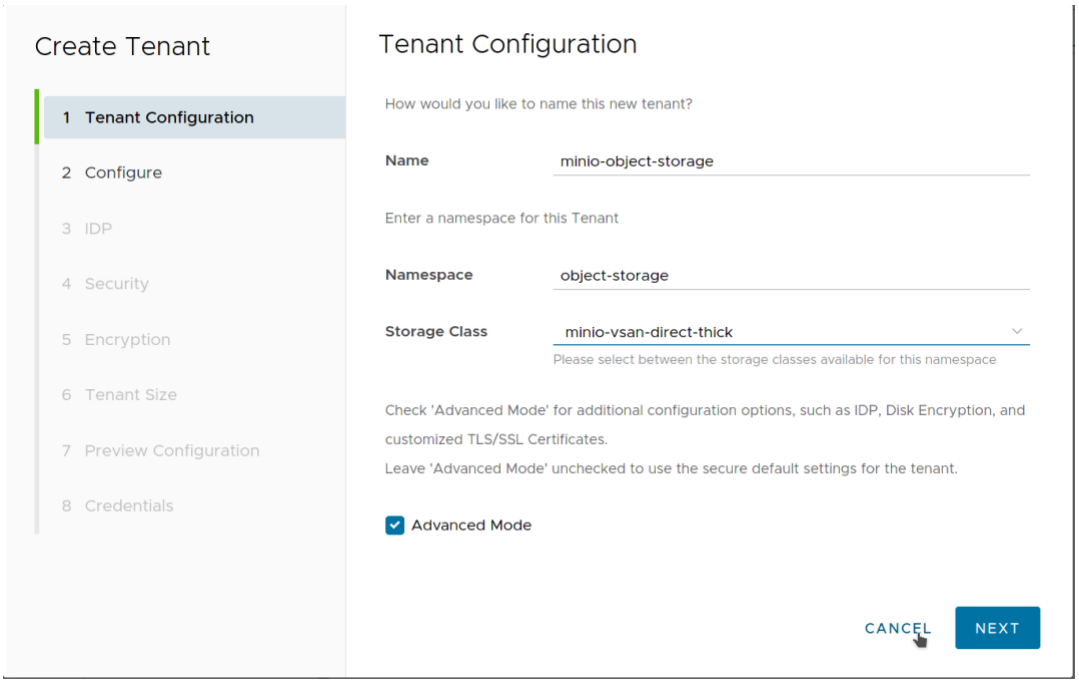
Step 1: Open the Tenant Creation Modal

Select the Datacenter and Cluster in which you want to deploy the MinIO tenant. Select the **Configure** tab, then expand the **MinIO** section. Select **Tenants**, then click **Add** to open the **Create Tenant** modal:



Step 2: Complete the Tenant Configuration Step

The Create Tenant modal displays Tenant Configuration as the first step:



- For **Name**, enter the preferred name of the MinIO Tenant.
- For **Namespace**, enter the Kubernetes namespace in which to deploy the MinIO tenant. The namespace must not contain any other MinIO tenants.
- For **Storage Class**, specify a storage class associated with the namespace. MinIO strongly recommends using a storage class associated with vSAN Direct or vSAN SNA drives, where the drives are locally attached to the ESXi hosts on which the MinIO tenant will deploy.

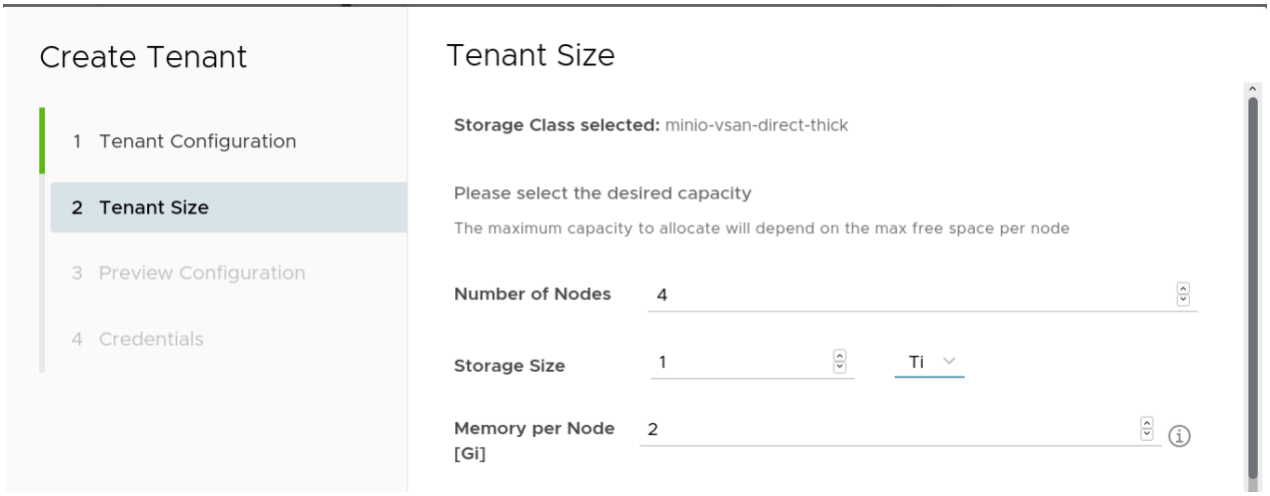
The **Advanced Mode** checkbox enables the following configuration steps:

- External Identity Providers (IDP)
- TLS Security
- Server-Side Encryption

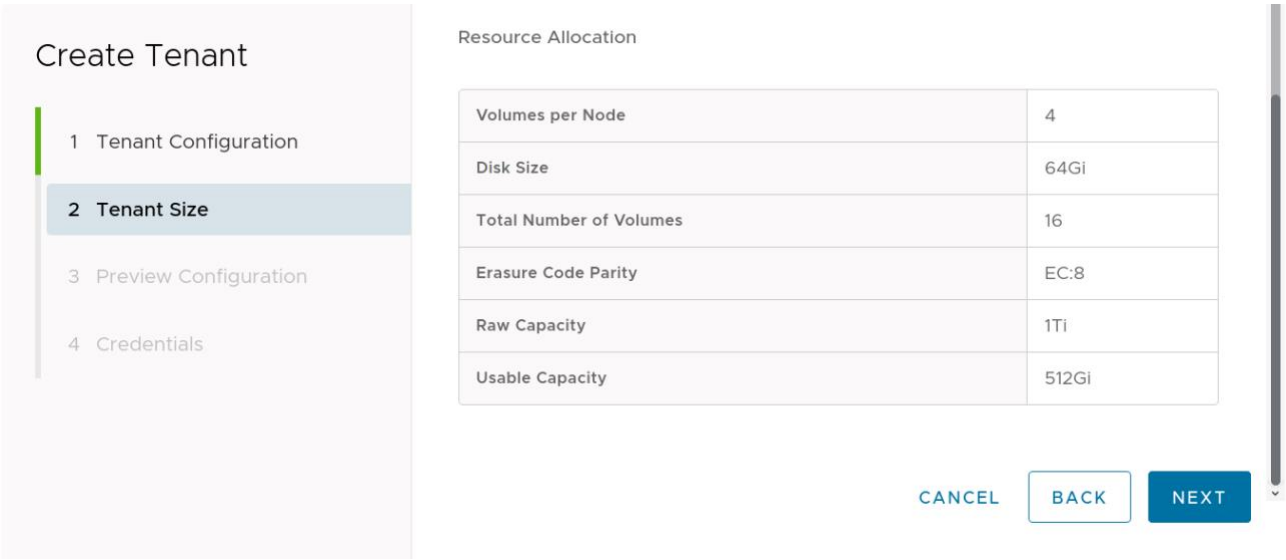
Advanced Mode features are not required for development and evaluation of MinIO on vSphere.

Step 3: Complete the Tenant Size Step

The **Tenant Size** step provides inputs for configuring the size and storage capacity of the cluster.



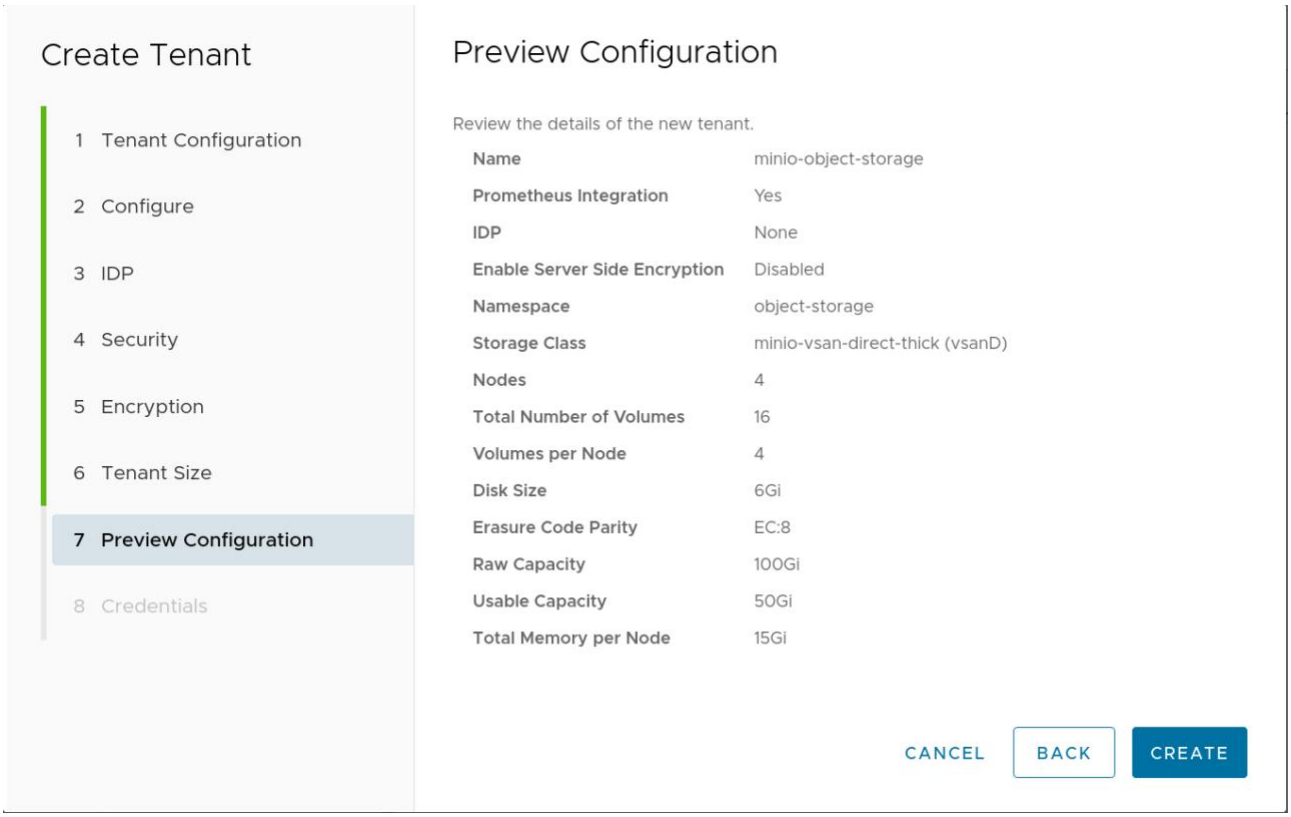
- For **Number of Nodes**, specify the number of MinIO Object Storage server pods to deploy in the tenant.
- For **Storage Size**, specify the total storage for the tenant. Since MinIO allocates a portion of storage for supporting Erasure Coding, the total storage must exceed the estimated total usable capacity required by applications. For example, at the highest Erasure Coding settings, the Storage Size must be *at least* 2 times the amount of expected usable storage.
- For **Memory per Node**, specify the amount of RAM to allocate to each MinIO Pod in the tenant.



The **Resource Allocation** section displays a summary of the specified configuration settings and the resulting intended tenant deployment settings.

Step 4. Complete the Preview Configuration Step

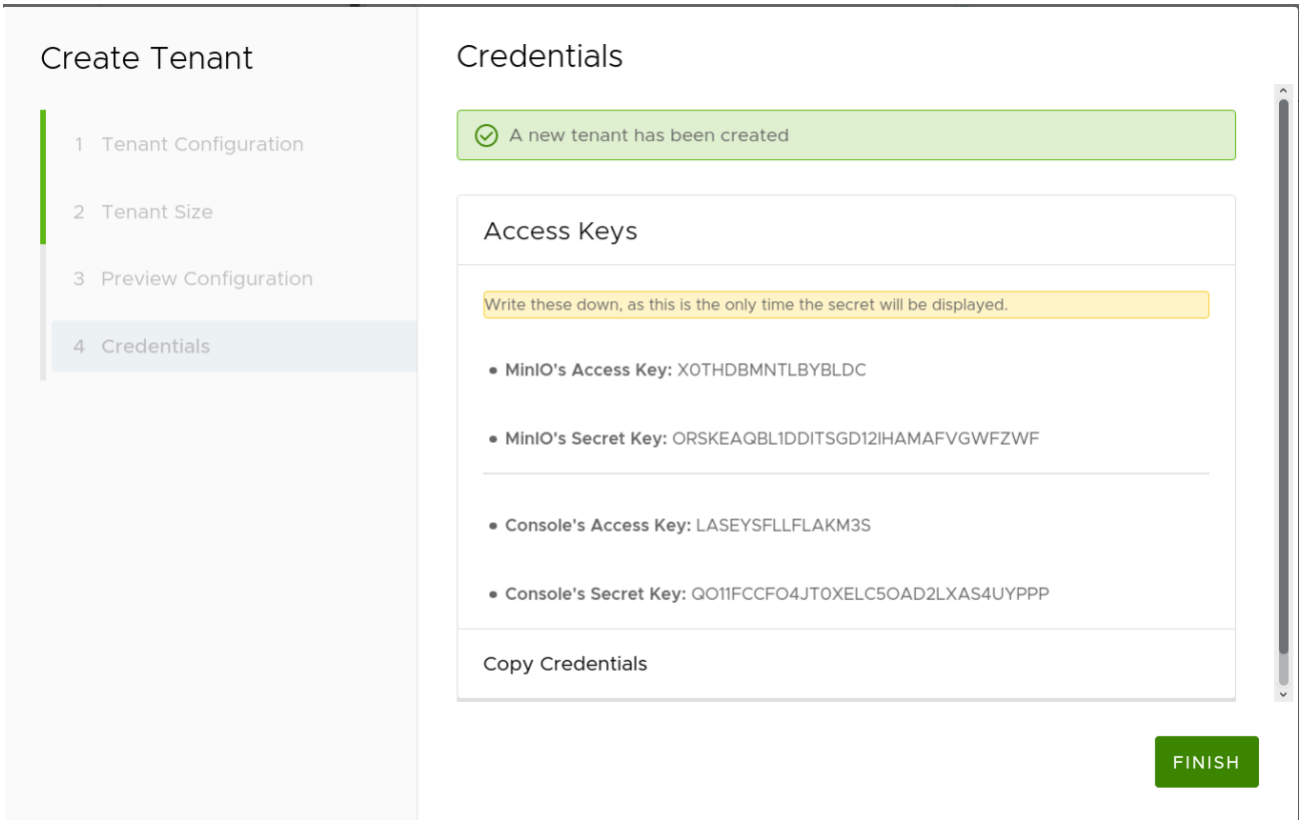
The **Preview Configuration** step summarizes the MinIO tenant configuration using the inputs from all previous steps. You can return to any previous step to modify the input parameters until the tenant meets the requirements.



Click **Create** to create the tenant and generate the access credentials.

Step 5. Complete the Credentials Step

The **Credentials** section displays credentials for connecting to MinIO tenant resources.



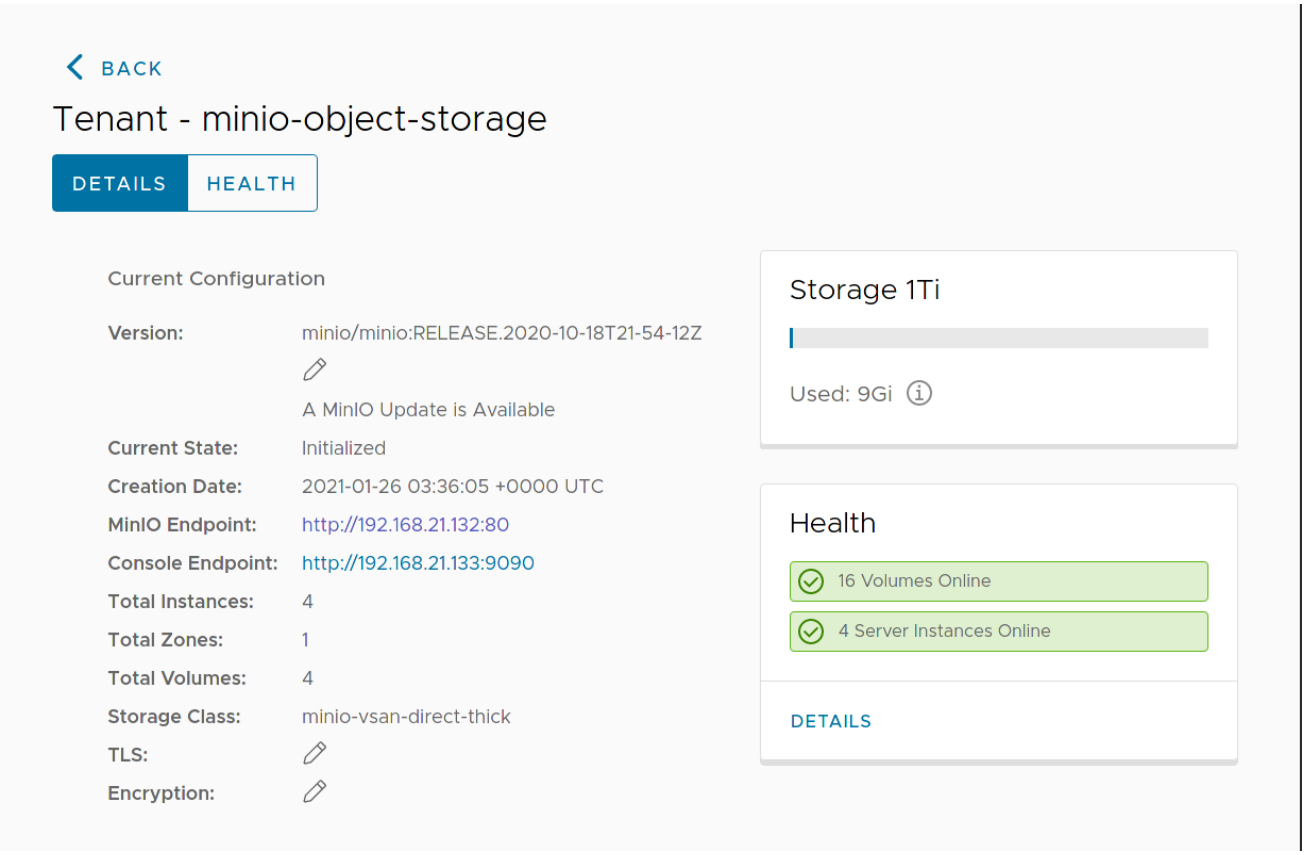
The **MinIO's Access Key** and **MinIO's Secret Key** provide administrative access to the MinIO Object Store. Applications may use these credentials for performing operations during early development and evaluation.

The **Console's Access key** and **Console's Secret Key** provide access to the MinIO Console, a browser-based interface for performing administrative actions on the MinIO tenant.

MinIO displays these credentials **exactly once**. If you do not copy down the credentials during this step, you cannot access the MinIO tenant. MinIO strongly recommends storing the credentials in a secure, password-protected location.

Step 6. View the Tenant

You can view the tenant details by clicking **Tenants** on the **MinIO** navigation item. When the **Current State** reads as **Initialized**, the tenant is ready for normal operations.



Failure Scenarios

MinIO protects data with per-object inline erasure coding written in assembly code to deliver the highest possible performance. MinIO uses Reed-Solomon code to stripe objects into data and parity blocks with user-configurable redundancy levels. MinIO's Erasure Coding performs healing at the object level and can heal multiple objects independently.

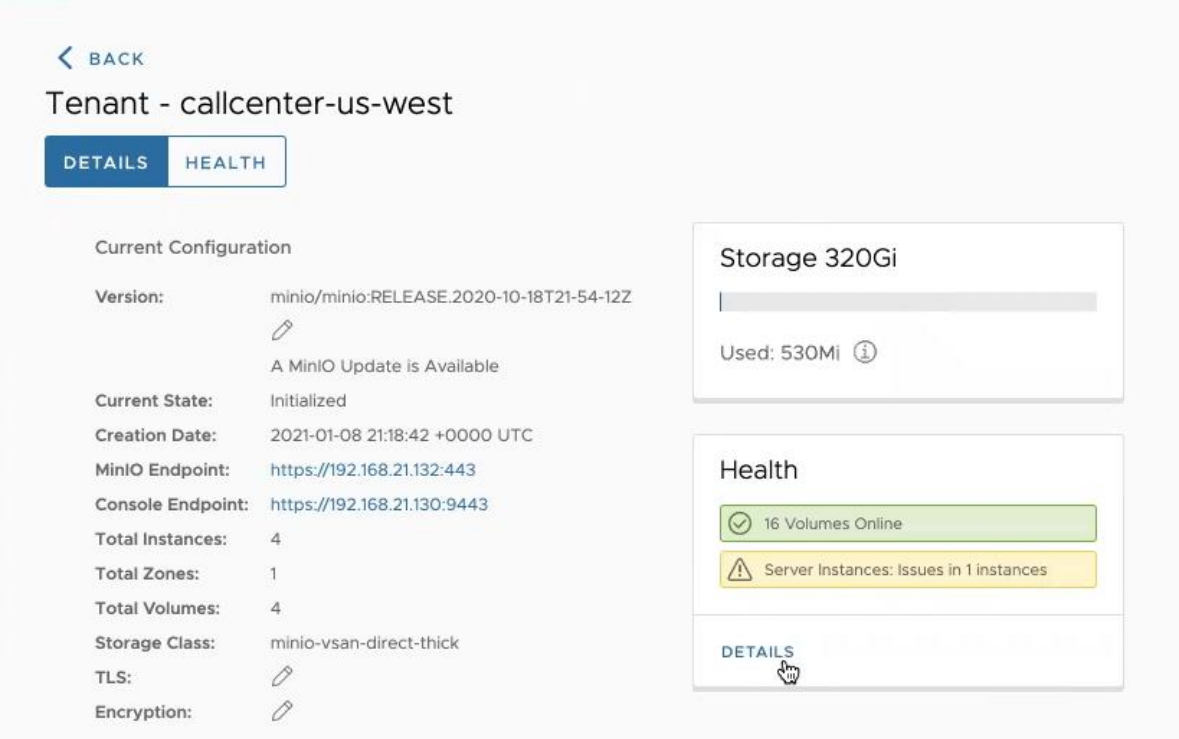
At the maximum parity of N/2 (N=Total Volumes), MinIO's implementation can ensure uninterrupted read and write operations with only ((N/2)+1) operational drives in the deployment. For example, in a 12-drive setup, MinIO shards objects across 6 data and 6 parity drives and can reliably write new objects or reconstruct existing objects with only 7 drives remaining in the deployment

While Erasure Coding healing is automatic and enabled by default, long-term resolution of the following failures requires user intervention:

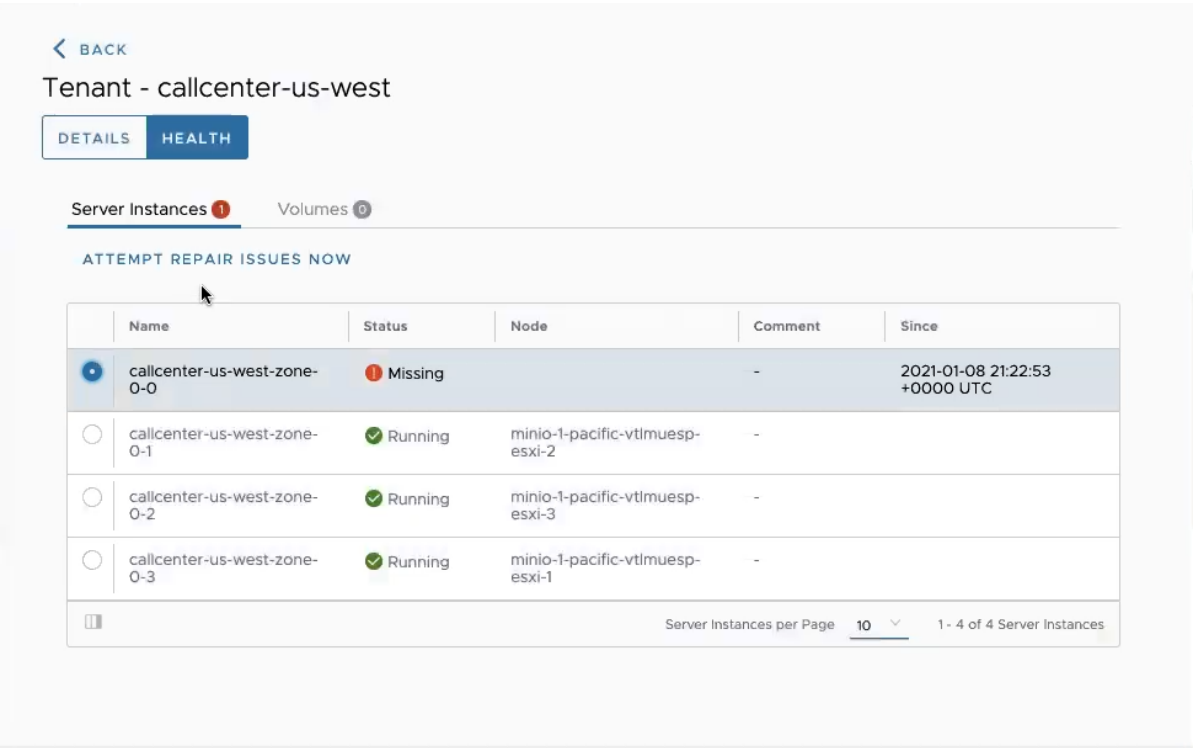
- Server failure, such as a downed ESXi host or a locked pod.
- Volume failure, such as a physical drive failure or bad Persistent Volume configuration.

MinIO provides built-in repair tools for resolving these scenarios directly through the vSphere interface.

Each MinIO tenant reports the status of its own health, including server and drive status:

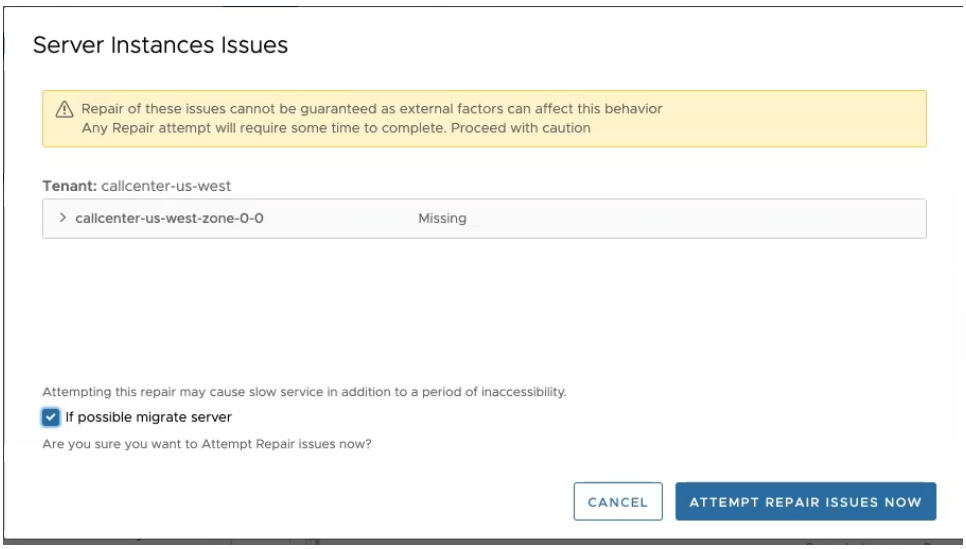


Under the **Health** display, click **Details** to review the current status of the tenant. You can also click the **Health** tab under the name of the tenant.



The **Server Instances** and **Volumes** tabs display health information on the servers and volumes supporting the tenant. MinIO displays the number of server or drive related issues in a red circle to the right of the tab. The **Status** column displays the specific problem, while the **Since** column displays when the problem occurred.

MinIO can attempt to repair server or volume level issues. Click the radio button on the row of the server or volume which requires repair and click the **Attempt Repair Issues Now** button to open the repair modal:



The modal displays a summary of the issue. You can click on the row to view more details on the problem server of volume.

Click **Attempt Repair Issues Now** to direct MinIO to attempt repair procedures.

- For server-level issues, MinIO attempts to restart the pod. If the pod is on a downed ESXi node,
- For volume-level issues, MinIO attempts to create new persistent volume claims for the purpose of replacing the bad volumes.

MinIO also offers the **If possible migrate server** checkbox. If selected *and* other repair attempts fail, MinIO migrates the MinIO server pod and all related volumes to a new available ESXi host with sufficient storage and compute resources. This option may be necessary if the existing ESXi host is no longer available and there are available nodes in the cluster. For volume-level repairs, server migration may be required for if the pod was using vSAN Direct or vSAN SNA hosts without sufficient existing capacity.

MinIO licensees have 24/7 access to the MinIO Subscription Network (SUBNET) for first-party support from MinIO engineering. Users can always raise issues on SUBNET for assistance and guidance on tenant repair operations.

NOTE: Repairs may require a significant amount of time to complete depending on the amount of data (if any) to migrate or heal as part of the repair. Repairs can also make the cluster briefly unavailable since the new server instance might need to be created from scratch.

Sizing Guidelines

The following recommendations provide the best practices and sizing guidance to run MinIO Object Storage on VMware Cloud Foundation using the vSAN Data Persistence platform.

VMware Cloud Foundation Infrastructure

Follow the general sizing guide: *Cloud Foundation Kubernetes Sizing Guide*

Compute Sizing

MinIO is hardware agnostic and runs on a variety of hardware architectures ranging from ARM-based embedded systems to high-end x64 and POWER9 servers. As a general guideline, memory should increase as the total per-host storage increases, while vCPU should increase as host NIC bandwidth increases.

Table 5.1 provides general guidelines for allocating memory for use by MinIO pods running on an ESXi host based on the total amount of storage supported by that pod:

Table 5.1. Memory Sizing

Total Host Storage	Host Memory (Minimum)
Up to 1 Tebibyte (Ti)	8GiB
Up to 10 Tebibyte (Ti)	16GiB
Up to 100 Tebibyte (Ti)	32GiB
Up to 1 Pebibyte (Pi)	64GiB
Greater than 1 Pebibyte (Pi)	128GiB

Table 5.2 provides general guidelines for allocating vCPU for use by MinIO pods running on an ESXi host based on the total network bandwidth supported by that pod.

Table 5.2. CPU Sizing

Host NIC Bandwidth	Host vCPU (Minimum)
10GbE or less	8 vCPU
25GbE	16 vCPU
50GbE	32 vCPU
100GbE	64 vCPU

IMPORTANT: vSphere defaults MinIO pods to 1vCPU. You can modify the number of vCPU allocated to each MinIO pod after deploying the tenant by doing the following:

1. Download and install the *VMware Commandline Tools*
2. Use kubectl vsphere login to create a context for accessing the VCF cluster. The vSphere user specified to kubectl vsphere login must have permission to access and perform operations on the namespace in which the tenant is deployed. See *Connecting to vSphere with Tanzu Clusters* for more specific instructions.
3. Run the following command to modify the vCPU allocation for the tenant:

   ```
   kubectl -n <NAMESPACE> patch tenant <TENANT-NAME> --type='json' -p='[{"op": "replace", "path": "/spec/zones/0/resources/limits/cpu", "value":<CPU-LIMIT>},{"op": "replace", "path": "/spec/zones/0/resources/requests/cpu", "value":<CPU-LIMIT>}]'
   ```

   Replace <NAMESPACE> with the name of the namespace in which the tenant is deployed.
   Replace <TENANT-NAME> with the name of the MinIO tenant

Replace <CPU-LIMIT> with the number of vCPU to allocate to the tenant.

For tenants with multiple zones, re-issue the command and increment /spec/zones/0 for each zone in the tenant.

Network Sizing

MinIO recommends high speed networking to support the maximum possible throughput of the storage (aggregated drives, storage controllers, and PCIe busses). The following table provides general guidelines for the maximum storage throughput supported by a given NIC.

Table 6. Network Sizing Guidelines

NIC bandwidth (Gbps)	Estimated Aggregate Storage Throughput (GBps)
10GbE	1GBps
25GbE	2.5GBps
50GbE	5GBps
100GbE	10GBps

MinIO is typically I/O bound at the network layer, where the maximum per-host throughput is constrained by the network infrastructure. For example, an ESXi host with 16 SSD SATA drives capable of ~200MBps has a maximum aggregate storage throughput of ~3.2GBps and therefore requires *at least* a 50GbE NIC and supporting infrastructure.

Storage Sizing

When sizing storage for a MinIO tenant, you must allocate more than the planned total usable capacity. MinIO uses Erasure Coding, an automatic data redundancy and availability feature that supports on-the-fly reconstruction of objects despite the loss of drives or nodes in the tenant. Erasure Coding uses parity blocks to support object healing, where higher parity levels provide increased redundancy and availability at the cost of total usable storage. MinIO uses the EC:N notation to refer to the number of parity blocks to create for a given object.

To calculate the usable ratio of storage, you must first calculate the size of each Erasure Set in the tenant zone. MinIO automatically creates Erasure Sets of uniform size, between 4 and 16 drives each. MinIO uses the greatest common denominator within that range for all drives in the Tenant to calculate the Erasure Set size. For example, a 16-drive tenant zone would consist of a single 16-drive Erasure Set. A 20-drive tenant Pool would consist of two 10-drive Erasure Sets.

Once you calculate the Erasure Set size, you can determine the usable storage ratio for a given parity level by using the following formula:

(ERASURE_SET_DRIVES - EC:N) / ERASURE_SET_DRIVES

The following table lists the calculated usable storage ratio for a given parity setting on a 16-drive tenant Pool

Table 7. Storage Sizing Guidelines

Erasure Code Parity	Total Data Drives	Total Parity Drives	Usable Storage Ratio
EC: 8	8	8	0.500
EC: 6	10	6	0.625
EC: 4 (Default)	12	4	0.750
EC: 2	14	2	0.875

Erasure Code Parity Selection

MinIO Erasure Coding splits objects into data and parity blocks based on the size of the Erasure Set. MinIO uses parity blocks to automatically heal damaged or missing data blocks when reconstructing an object. MinIO uses the EC:N notation to refer to the number of parity blocks (N) in the Erasure Set.

Erasure Code Parity dictates the number of drives tenant Pool can tolerate the loss of while still serving read and write operations. Generally, higher levels of parity tolerate more drive loss and the cost of total usable storage. Specifically:

- Read Quorum: The minimum number of drives required to serve read operations on an object. Read Quorum is calculated as: ERASURE_SET_DRIVES-(EC:N)
- Write Quorum: The minimum number of drives required to serve write operations. Write Quorum is calculated as: ERASURE_SET_DRIVES - (EC:N-1)

The following table lists the total usable storage, storage ratio, read quorum, and write quorum for a 16-drive MinIO tenant Pool using a given parity setting:

Table 8. Parity Selection Guidelines

Parity	Total Storage	Storage Ratio	Read Quorum (Minimum Drives)	Write Quorum (Minimum Drives)
EC: 8 (Default)	8 Tebibytes	0.500	8	9
EC: 6	10 Tebibytes	0.625	10	11
EC: 4	12 Tebibytes	0.750	12	13
EC: 2	14 Tebibytes	0.875	14	15

MinIO supports two storage classes for per-object parity levels. STANDARD parity defaults to EC:(ERASURE_SET_DRIVES/2). REDUCED parity defaults to EC:2.

Use Cases

Given MinIO's performance and scalability attributes, it can address a broad range of use cases, resulting in superior return on investment. These use cases can be categorized as follows:

Backup/Snapshots/Archival – often considered the core object storage use cases, MinIO brings best in class performance to these tasks, ensuring that both back and restore are done so accurately and quickly. Examples include Veeam, Commvault and Rubrik.

AI and ML - This is a new class of use cases for object storage and is enable by the combination of performance and scale that MinIO offers. In these examples, MinIO is the datastore for ML and AI data pipelines where training, inference reads/writes occur. Examples include TensorFlow, H2o.ai, Spark, Presto and Flink.

Web/Mobile Cloud Native Application Development - Object storage is the storage class of the cloud. As a result, it is the default storage type for the tens of thousands of internally developed applications (consumer and enterprise) where performance and scale are important. By offering a RESTful API, MinIO is designed for containerization and orchestration – just like the cloud native applications it supports. Examples include UnitedHealth, Box.com, Raytheon and Space X.

Analytics – The default architecture for modern analytics platforms is to use an S3 endpoint for your data. This is the ultimate test of performance at scale as many of these instances are 50 PB or more in size and carry the expectation of a "warm" tier at a minimum and "hot" tier in performance-oriented environments. Examples include Splunk SmartStores, Vertica EON mode and Teradata NOS.

Artifact Stores – The cloud native world has created a new class of storage use cases, that of artifact storage. These instances can grow quickly and require high throughput, making them excellent for MinIO. Examples include Docker Registries like Harbor, Cloud Foundry Artifact Stores and applications like Concourse.

Conclusion

MinIO is an important part of VMware's Kubernetes architecture on VMware Cloud Foundation with Tanzu. While the vision is expansive, MinIO is able to bring distinct capabilities and functionality to the vSAN Data Persistence platform ecosystem. The result is performant, persistent storage and a host of cloud native applications that will be turnkey for VMware's partners and customers.

This is exciting opportunity to extend our partnership between MinIO and VMware to support this extraordinary journey that unites IT admins and DevOps and accelerates the adoption of Kubernetes in the enterprise, consistently offering simplicity, superior manageability, and ultimately lower total cost of ownership, whether in the private cloud or the public cloud.

Reference

- *VMware Cloud Foundation*
- *VMware vSphere*
- *VMware vSAN*
- *VMware NSX Data Center*
- *VMware vRealize Suite*
- *MINIO*
- *Document for MinIO on VMware Cloud Foundation*

About the Author

This paper was co-authored by the following people:

- Ka Kit Wong, Staff Solutions Architect, Solutions Architecture team of the Cloud Platform Business Unit at VMware
- Ting Yin, Senior Solutions Architect, Solutions Architecture team of the Cloud Platform Business Unit at VMware
- Ravind Kumar, Lead Technical Writer at MinIO

The following reviewers also contributed to the paper contents:

- César Nieto, Software Engineer at MinIO
- Lenin Alevski Huerta, Security Software Engineer at MinIO
- Ritesh H Shukla, Engineering at MinIO

VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com.