Install MinIO 2.0.10 on vSphere 8.0

November 2026

2.0.10 is the latest certified release and the MinIO version VMware lists in its Supervisor Services catalogue (v2.0.10 changelog entry).

Service definition	minio-service-definition-2.0.10.yaml: the one originally released (sha256 b9f8193c…), with registryName as a registry and project (changelog)
sha256	e6b4ca86a04d229eeb69f094373faf9bdfbbc330e7699a4ba82507b3e9a4e1c3
Operator	quay.io/minio/operator:v5.0.16 (public)
Plug-in	quay.io/miniodev/vsphere:v2.0.10 (public)

Prerequisites, the YAML Service Config keys and removal are in vsphere-8-reference.md.

Before you start: copy the images to a private registry

Required. The Supervisor pulls the operator and plug-in images from a private registry of yours: copy them into one project of that registry before the install, and give the Supervisor that registry's credentials (step 2). registryName is that registry **and project**, e.g. registry.example.com/minio, and the definition pulls the images from it under their own names:

Image	Source	In the private registry (registryName: registry.example.com/minio)
Operator	quay.io/minio/operator:v5.0.16	registry.example.com/minio/operator:v5.0.16@sha256:…
Plug-in	quay.io/miniodev/vsphere:v2.0.10	registry.example.com/minio/vsphere:v2.0.10@sha256:…
Tenants' default MinIO image	minio/minio:RELEASE.2024-07-04T14-25-45Z, **no longer provided**	registry.example.com/minio/minio:RELEASE.2024-07-04T14-25-45Z

1. **Create the project** in the private registry (in Harbor, e.g. minio) and a robot account that can pull from it.
2. **Copy the operator and plug-in images** into the project, with all their platforms, so that the digests the definition pins stay the same. Do not rebuild them. The 2.0.10 images are public on quay.io. With skopeo, logged in to the private registry:

```
PROJECT=registry.example.com/minio
skopeo copy --all docker://quay.io/minio/operator:v5.0.16 docker://$PROJECT/operator:v5.0.16
skopeo copy --all docker://quay.io/miniodev/vsphere:v2.0.10 docker://$PROJECT/vsphere:v2.0.10
```



Check that the digests the definition pins are there (also listed in CHANGELOG.md):

```
skopeo inspect --raw docker://$PROJECT/
operator@sha256:a2c1c7181e2fb5f5cc8d6d9d1d6ed44c03f282e6fe6234f42718cc4cd7f37f81 \
  >/dev/null && echo ok
skopeo inspect --raw docker://$PROJECT/
vsphere@sha256:60ff785b362789fa12e4903cf2cabb02cbdca9255b265e8045245a17b3677f44 \
  >/dev/null && echo ok
```

3. **The MinIO image for the tenants.**

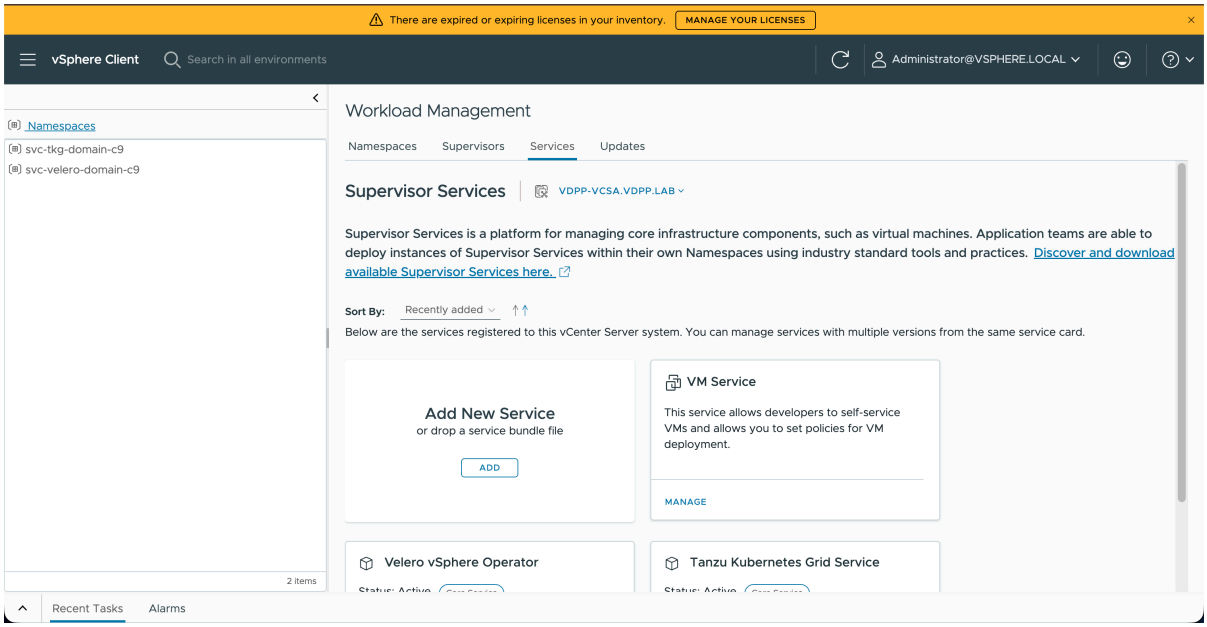
> **Warning: MinIO no longer provides the open-source minio/minio images.** Neither quay.io nor Docker Hub serves them any more, including the releases 2.0.10 tenants use (Known issues). To run them you need a copy of your own, e.g. minio/minio:RELEASE.2024-02-09T21-25-16Z: push it into the same project as $PROJECT/minio:<release>, and select it in the Create Tenant wizard with **Use custom image**.

The AIStor image, quay.io/minio/aistor/minio:RELEASE.2026-09-19T17-05-25Z, is public: copying it into the project is optional. It needs an AIStor license, which the plug-in manages from 2.3.0 (2.3.0 guide).

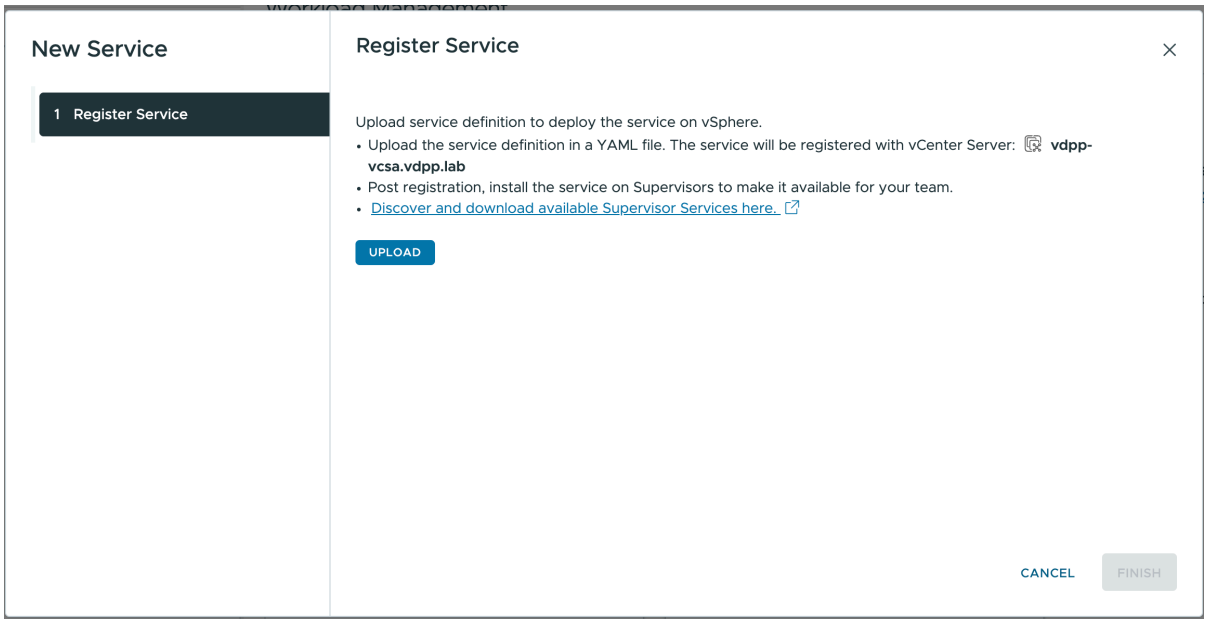
4. **The Supervisor must trust the registry's certificate.** A certificate from a private CA has to be added to the Supervisor before the install.

1. Register the service

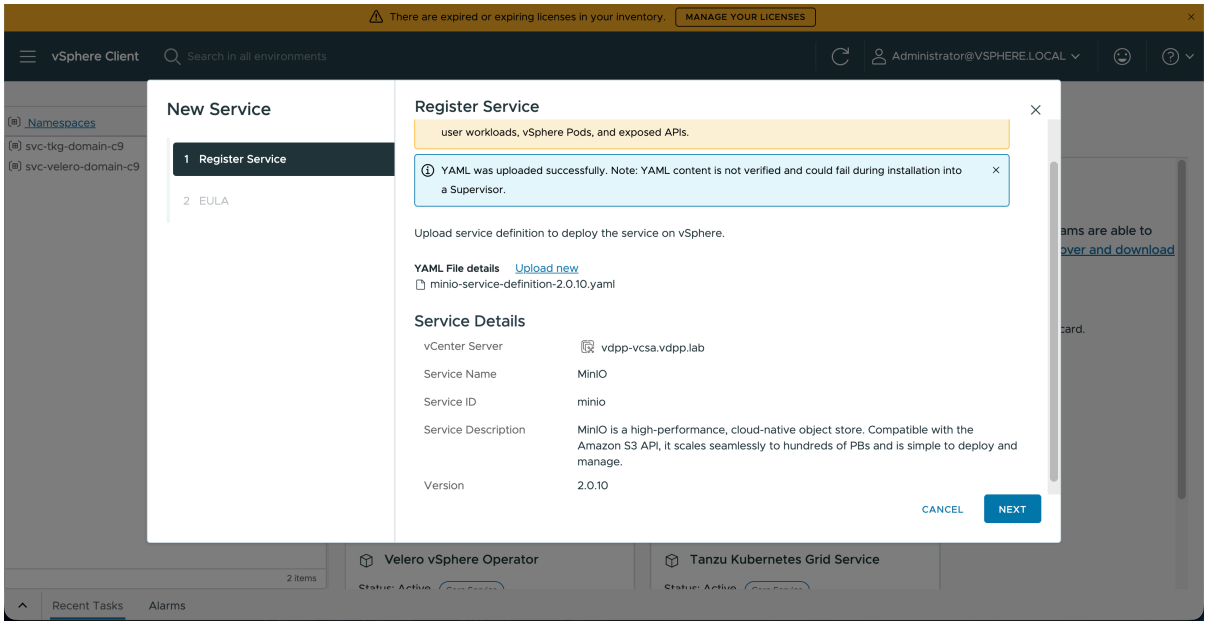
1. Check the definition: shasum -a 256 minio-service-definition-2.0.10.yaml must print the sha256 above.
2. In the vSphere Client, open **Menu → Workload Management → Services**.



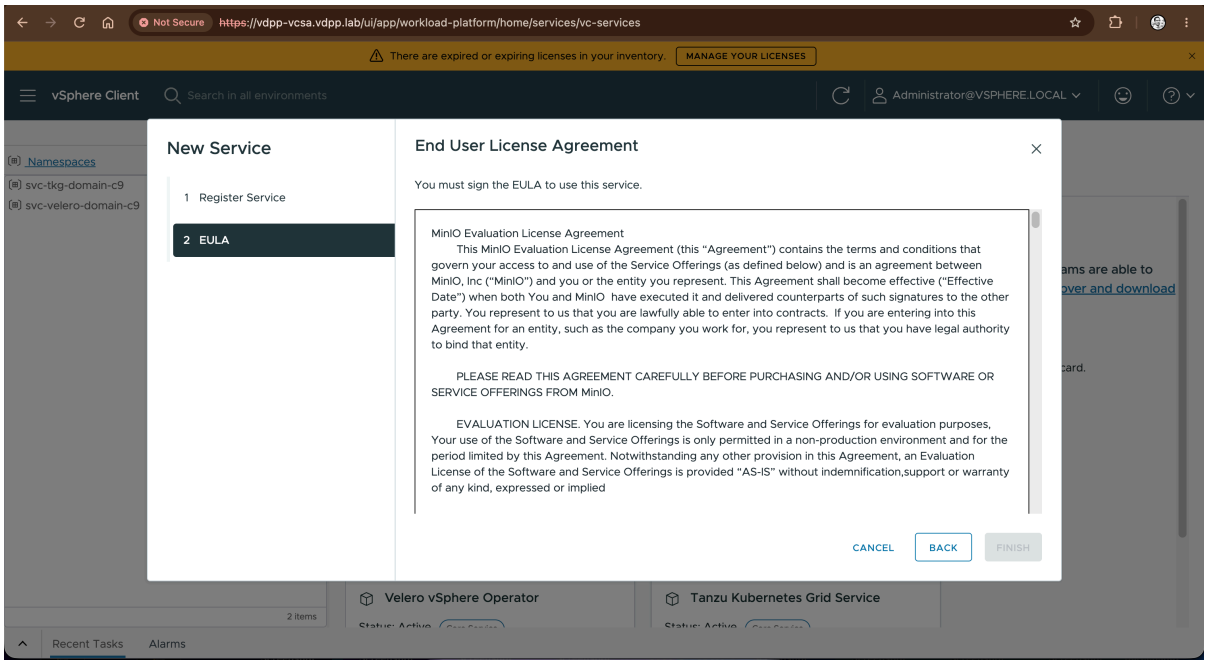
3. Click **ADD** on the **Add New Service** card, or drop the YAML file on it. The **New Service** wizard opens on **Register Service**.



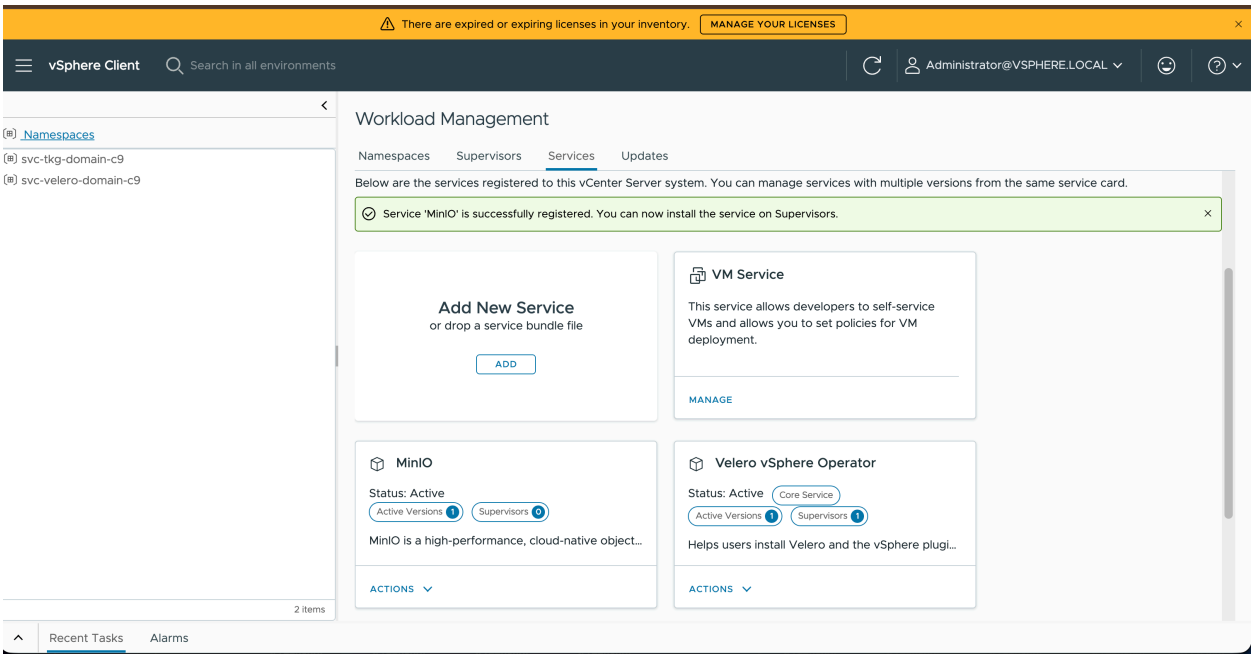
4. Click **UPLOAD** and select the YAML file. The step confirms the upload and shows **Service Details**: Service Name **MinIO**, Service ID **minio**, Version **2.0.10**. Click **NEXT**.



5. **EULA**: read the **MinIO Evaluation License Agreement**, accept it, and click **FINISH**.

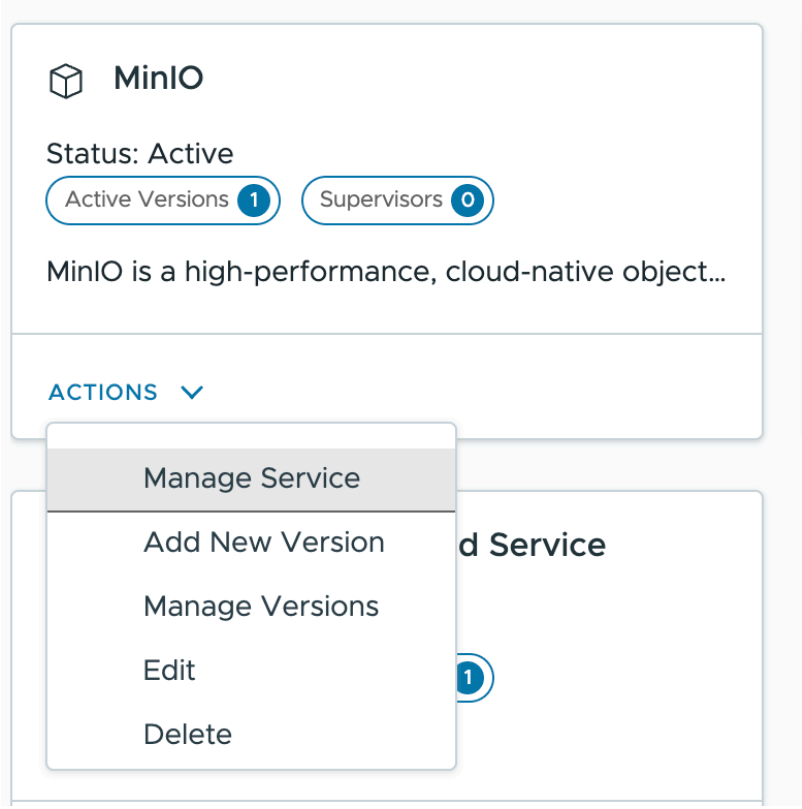


The Services page confirms **"Service 'MinIO' is successfully registered. You can now install the service on Supervisors."** and shows a **MinIO** card: **Status: Active**, **Active Versions 1**, **Supervisors 0**, and an **ACTIONS** menu. Unlike VMware's services, it has no **Core Service** badge. No svc-minio-… namespace exists yet: only the core services' svc-tkg-domain-c<N> and svc-velero-domain-c<N> are listed under **Namespaces**.



2. Install it on the Supervisor

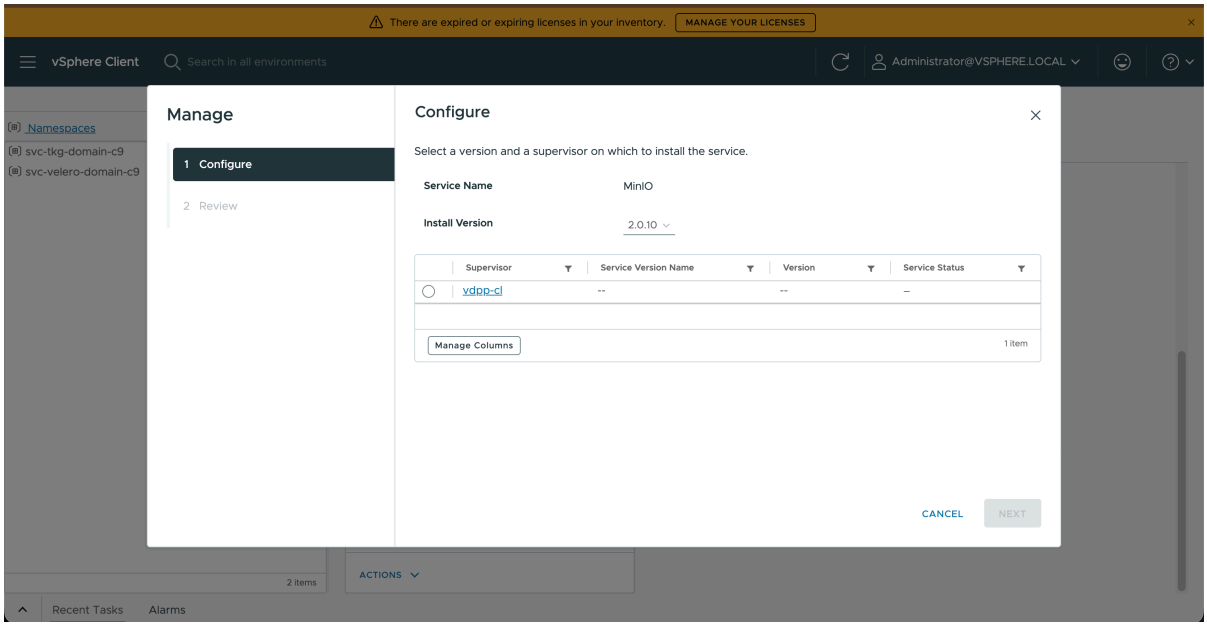
1. On the **MinIO** card, open **ACTIONS** → **Manage Service**. This is the entry that installs the service on Supervisors.



2. The **Manage** wizard opens on **Configure** ("Select a version and a supervisor on which to install the service"), followed by **Review**:

   - **Install Version**: **2.0.10**.
   - In the Supervisor table, select the radio button of the Supervisor (here vdpp-cl). Its **Service Version Name**, **Version** and **Service Status** columns are empty while MinIO is not installed on it.

   Click **NEXT**.



3. **Review** checks the pair and reports **"Selected service version and Supervisor are compatible."**, then lists **Service Name** MinIO, **Version to install** 2.0.10 and **Supervisor** vdpp-cl.

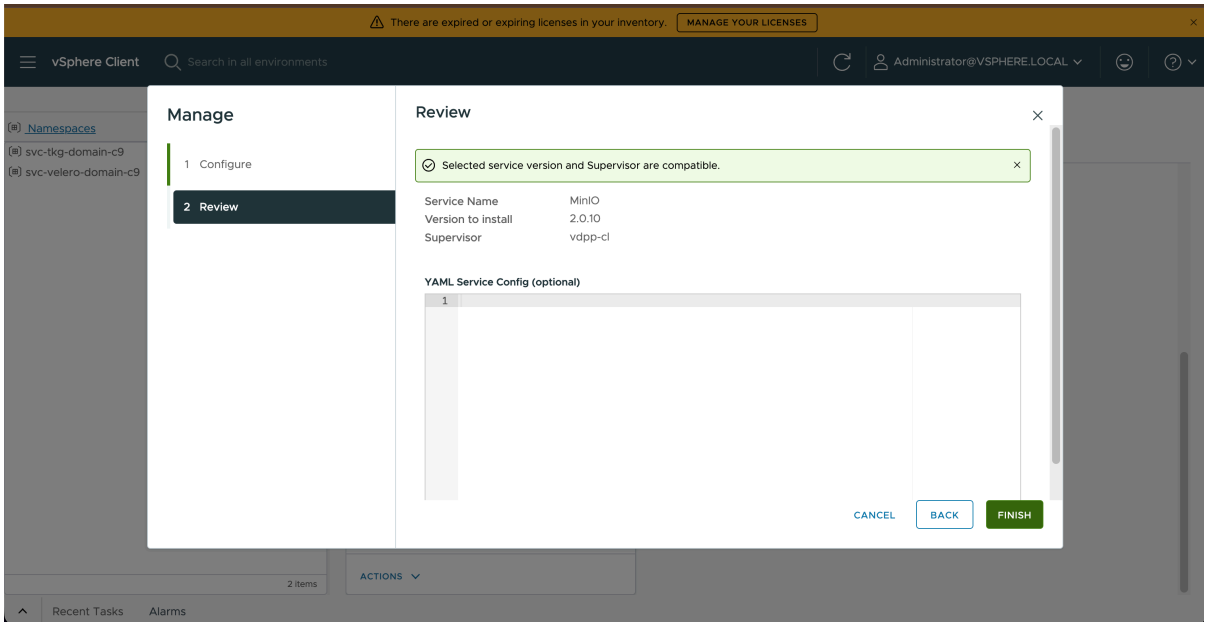
4. **YAML Service Config (optional)**, an editor on the same Review page: enter the private registry and project holding the copied images (copy the images), and its credentials:

```
registryName: registry.example.com/minio
registryUsername: <robot account>
registryPasswd: <its password>
```

> **Important: registryName must include the project** (registry.example.com/minio, not registry.example.com). The definition builds every image name as <registryName>/<image>: <registryName>/operator:v5.0.16, <registryName>/vsphere:v2.0.10 and <registryName>/minio:<release> for tenants. The images were copied into a project, and a registry such as Harbor keeps every repository in a project. With the host alone, the Supervisor looks for registry.example.com/operator, which does not exist, and the operator and plug-in pods stay in ErrImagePull.

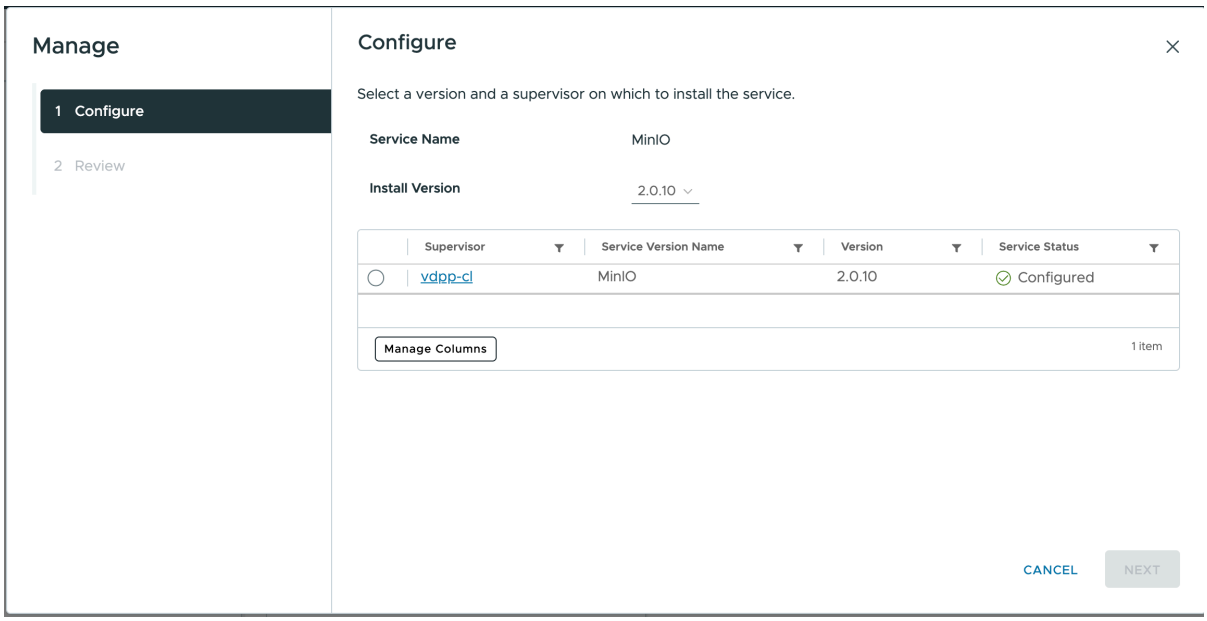
registryName is without https:// or a trailing /. The pull secret is created for the registry host, so it covers the project. The screenshot below shows a lab install with the YAML Service Config empty; the 2.0.13 guide shows it filled in.

5. Click **FINISH**.



3. Verify

1. **MinIO card → ACTIONS → Manage Service** (then **CANCEL**): the Supervisor row shows **Service Version Name** MinIO, **Version** 2.0.10, **Service Status Configured**. In the lab this took about a minute.



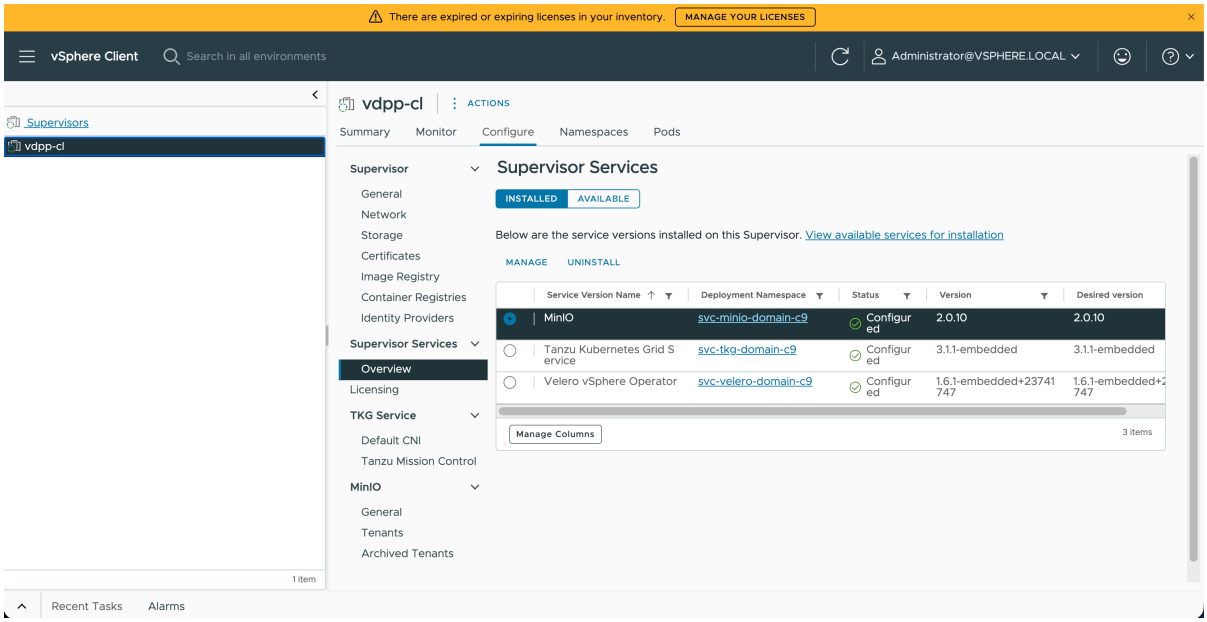
2. In the service namespace svc-minio-domain-c<N>:

```
kubectl get pods -n svc-minio-domain-c<N>
```

   Expected: …-minio-operator-…, …-vsphere-… and …-vsphere-controller-… **Running**, on <registryName>/operator:v5.0.16 and <registryName>/vsphere:v2.0.10, with the pinned digests.

3. Storage policies created by the service: minio-vsan-direct-thick and minio-vsan-sna-thick.

4. The MinIO plug-in appears in the vSphere Client after a reload (log out and in): **Workload Management → Supervisors → <supervisor> → Configure → MinIO**. On vSphere 8 it attaches to the **Supervisor** object, not to the cluster in Hosts and Clusters. vCenter registers it as io.min.vsphere (plug-in version 2.1.0.0, from the plug-in's own manifest). The Supervisor's **Configure** tab shows a **MinIO** group (**General**, **Tenants**, **Archived Tenants**), and **Supervisor Services → Overview** lists MinIO, namespace svc-minio-domain-c<N>, **Configured**, version and desired version **2.0.10**.



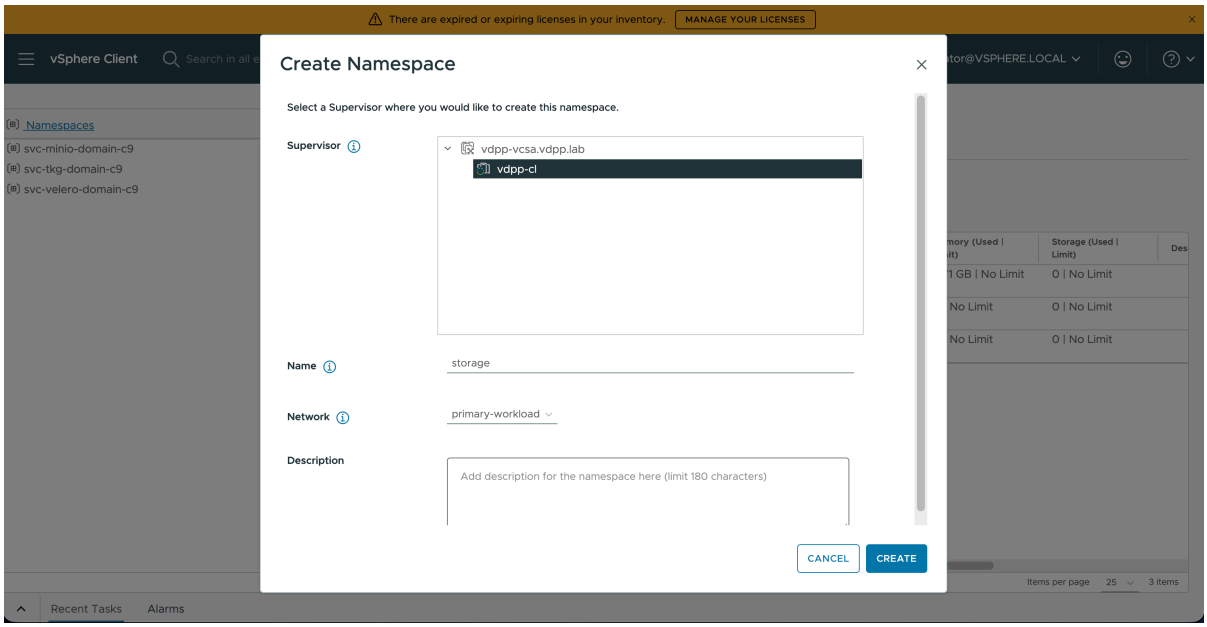
4. Prepare a namespace for tenants

A MinIO tenant runs in a vSphere Namespace that can use a vSAN Direct storage policy.

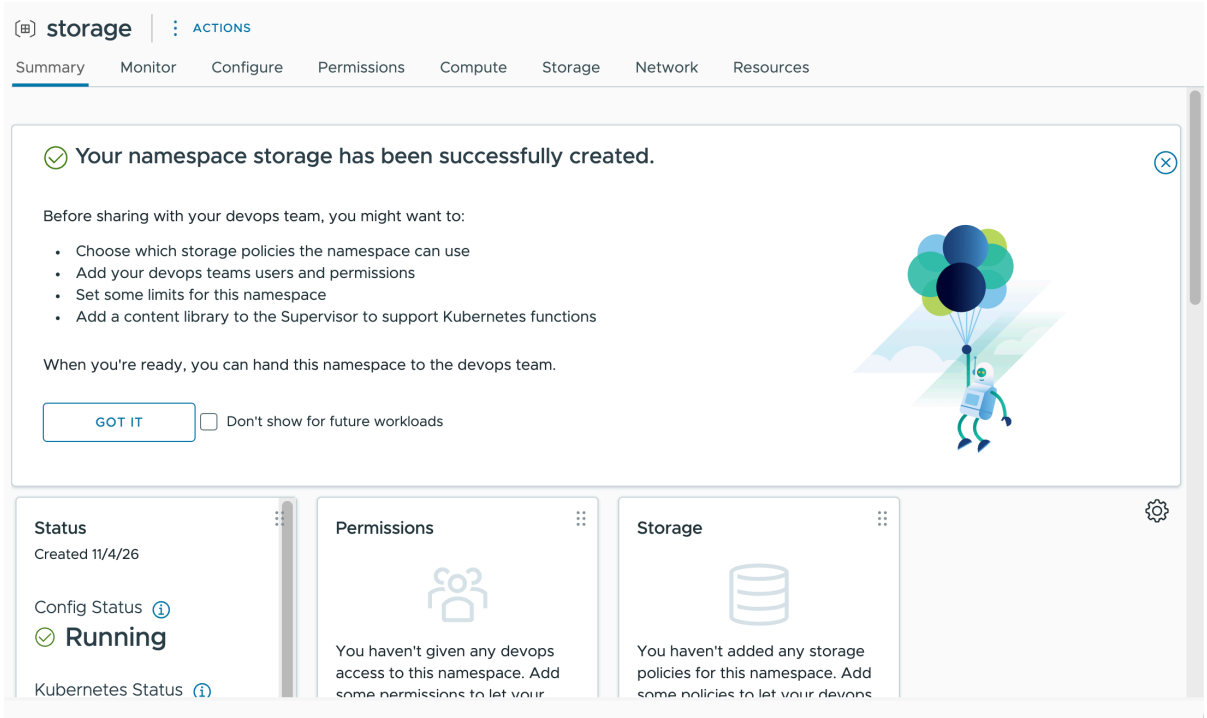
1. **Workload Management → Namespaces → NEW NAMESPACE**. In **Create Namespace**:

   - **Supervisor**: the Supervisor under its vCenter (here vdpp-cl under vdpp-vcsa.vdpp.lab);
   - **Name**: e.g. storage;
   - **Network**: primary-workload;
   - **Description**: optional, up to 180 characters.

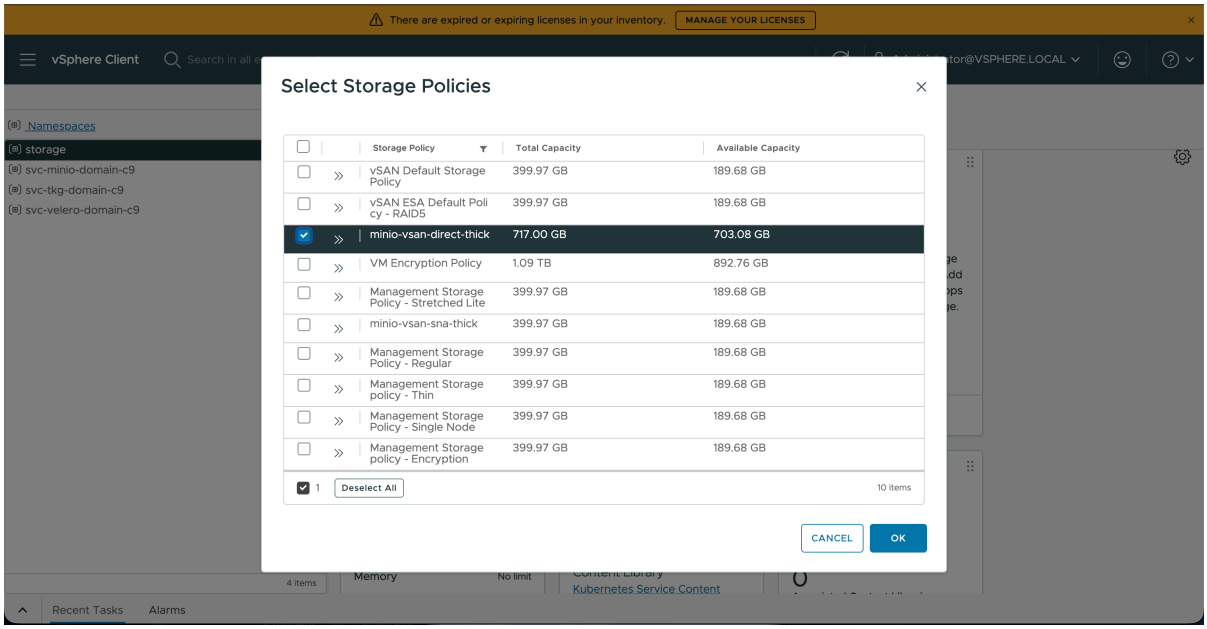
   Click **CREATE**.



2. The namespace opens on **Summary** with "Your namespace storage has been successfully created.", **Config Status: Running**, and empty **Permissions** and **Storage** cards.



3. **Storage** card (or the **Storage** tab) → **ADD STORAGE**. In **Select Storage Policies**, tick **minio-vsan-direct-thick** and click **OK**.

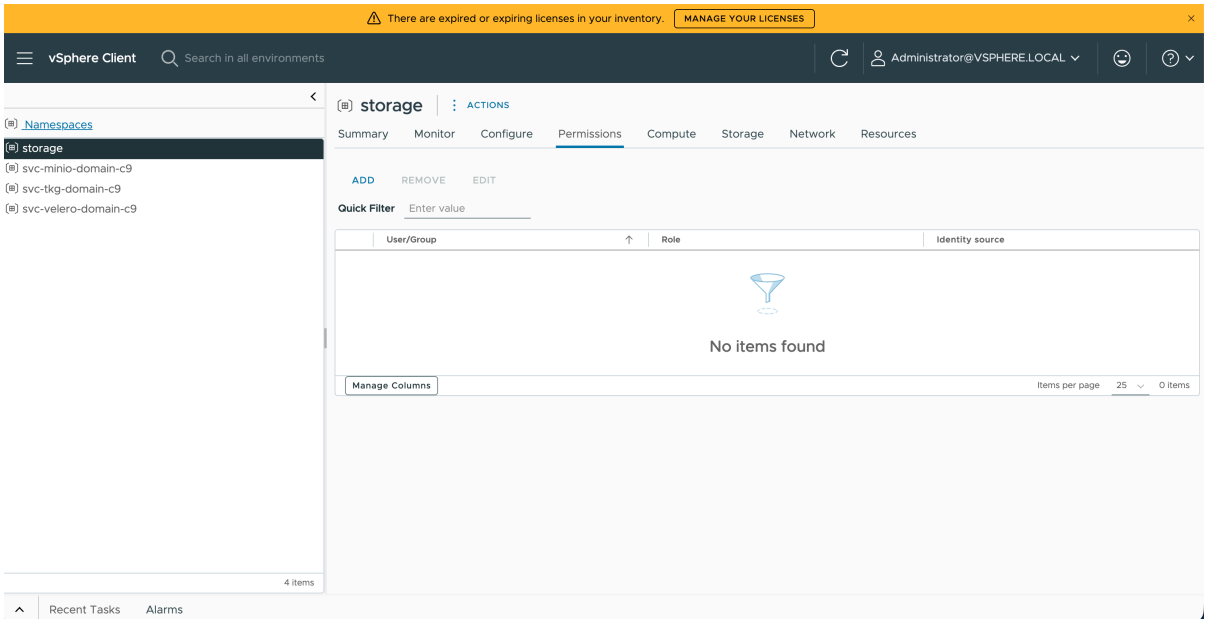


The two policies the MinIO service created differ in where the data goes:

Policy	Backed by	Lab capacity
minio-vsan-direct-thick	the **vSAN Direct** disks (one datastore per disk, no vSAN replication)	717 GB total
minio-vsan-sna-thick	the **vSAN** datastore, with shared-nothing placement	same as the vSAN datastore (≈400 GB)

vSAN Direct is the layout MinIO is meant for: MinIO's erasure coding provides the redundancy.

4. **Permissions** tab → **ADD** the users or groups who will create tenants, with **Can edit**. The list starts empty, but members of vsphere.local\administrators already have edit rights: vCenter creates the RoleBinding wcp:<namespace>:group:vsphere.local:administrators (ClusterRole edit) with the namespace. administrator@vsphere.local needs no entry here.



Adding the storage policy makes the Kubernetes StorageClass minio-vsan-direct-thick (binding mode WaitForFirstConsumer) usable in the namespace, through the ResourceQuota <namespace>-storagequota.

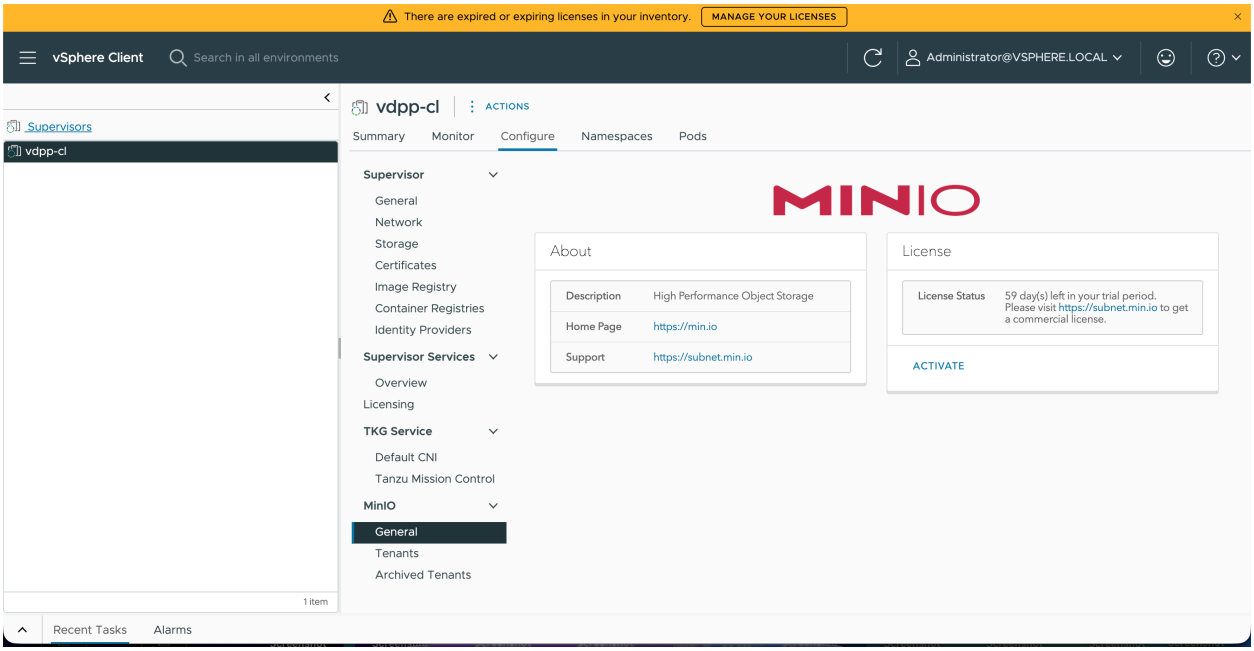
5. Create a tenant

All MinIO pages are under **Workload Management → Supervisors → <supervisor> → Configure → MinIO**.

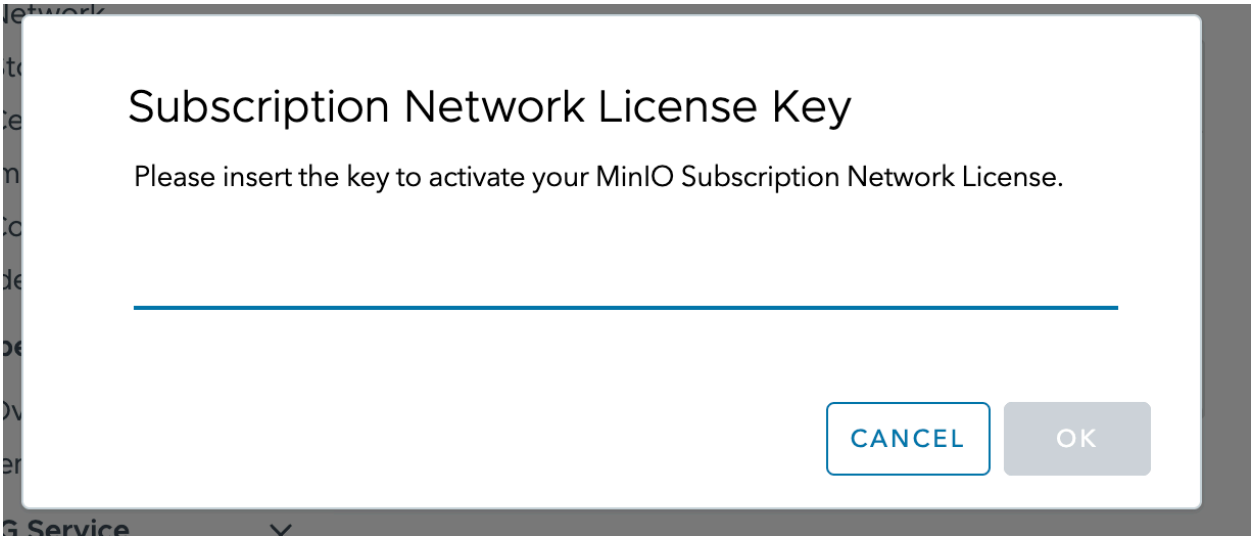
5.1 Check the license

MinIO → General shows **About** (description, home page, support) and **License**. Without a license, 2.0.10 runs a **60-day trial** counted from the creation of the minio-vsphere Deployment, so the day after the install it shows "59 day(s) left in your trial period". **ACTIVATE** takes a SUBNET license key, which the plug-in stores in the Secret minio-license-secret of the service namespace. The trial is enough to create tenants.

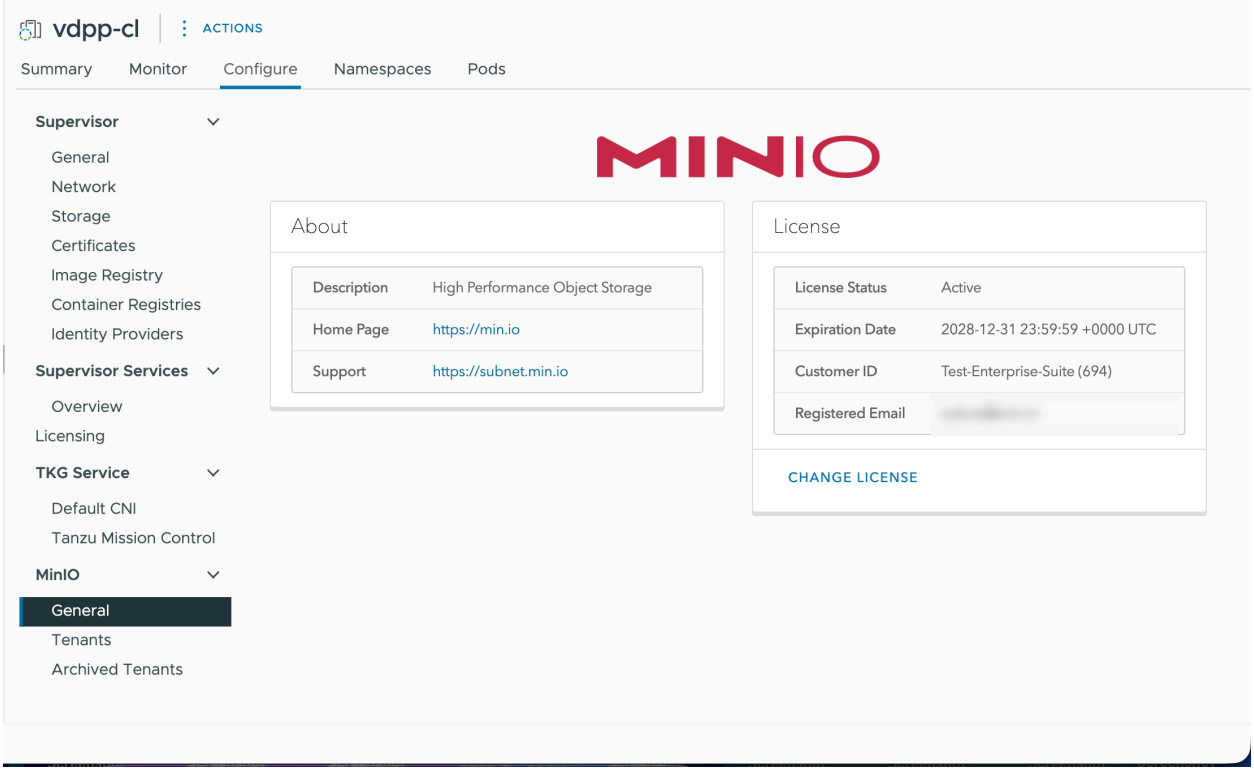
2.0.10 takes the older SUBNET license keys. Newer releases only accept AIStor licenses and have no trial.



ACTIVATE opens **Subscription Network License Key** ("Please insert the key to activate your MinIO Subscription Network License."); **OK** is enabled once a key is entered. The key is verified against SUBNET's public key inside the plug-in, so activation needs no Internet access from the Supervisor.



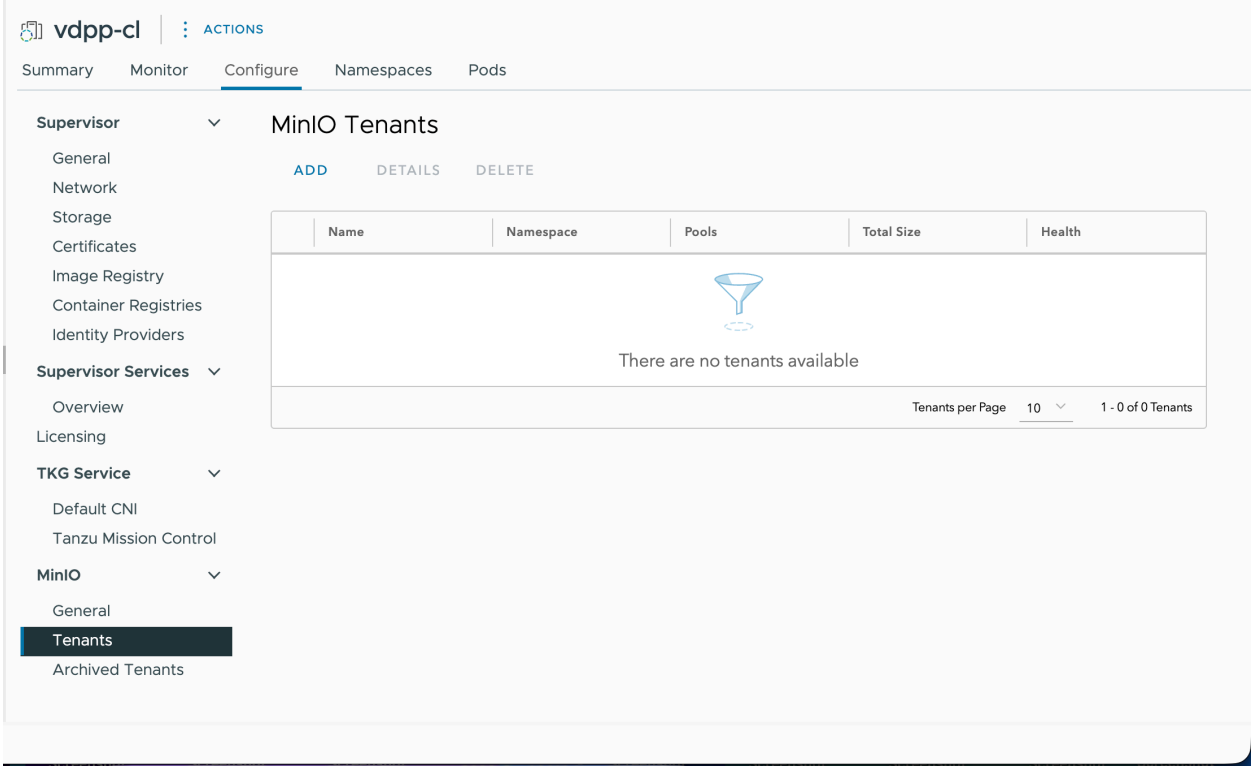
After **OK**, General shows **License Status: Active**, the **Expiration Date**, **Customer ID** and **Registered Email** from the license, and the button becomes **CHANGE LICENSE**:



If **License Status** shows an error instead, the plug-in cannot reach the Supervisor's Kubernetes API; see the plug-in pods' logs in svc-minio-domain-c<N>.

5.2 Add the tenant

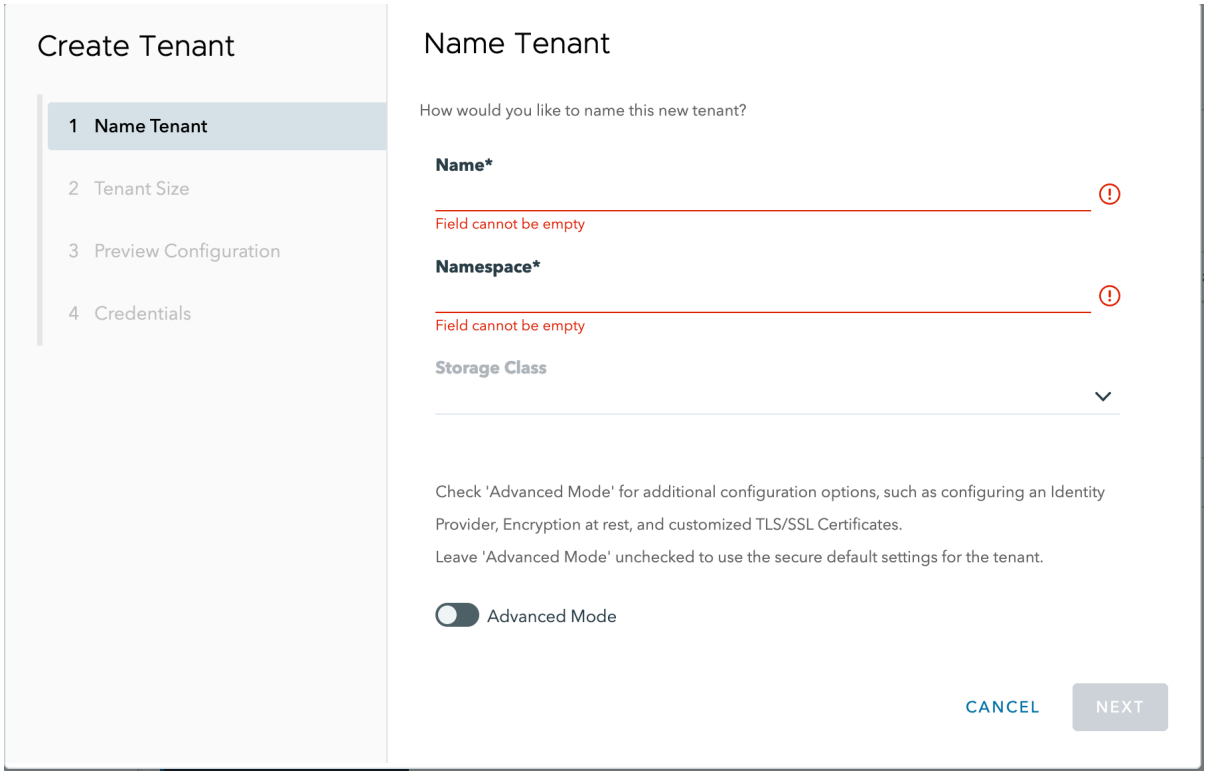
MinIO → Tenants lists the tenants of every namespace — **Name**, **Namespace**, **Pools**, **Total Size**, **Health** — with **ADD**, **DETAILS** and **DELETE**. It starts with "There are no tenants available". Click **ADD**.



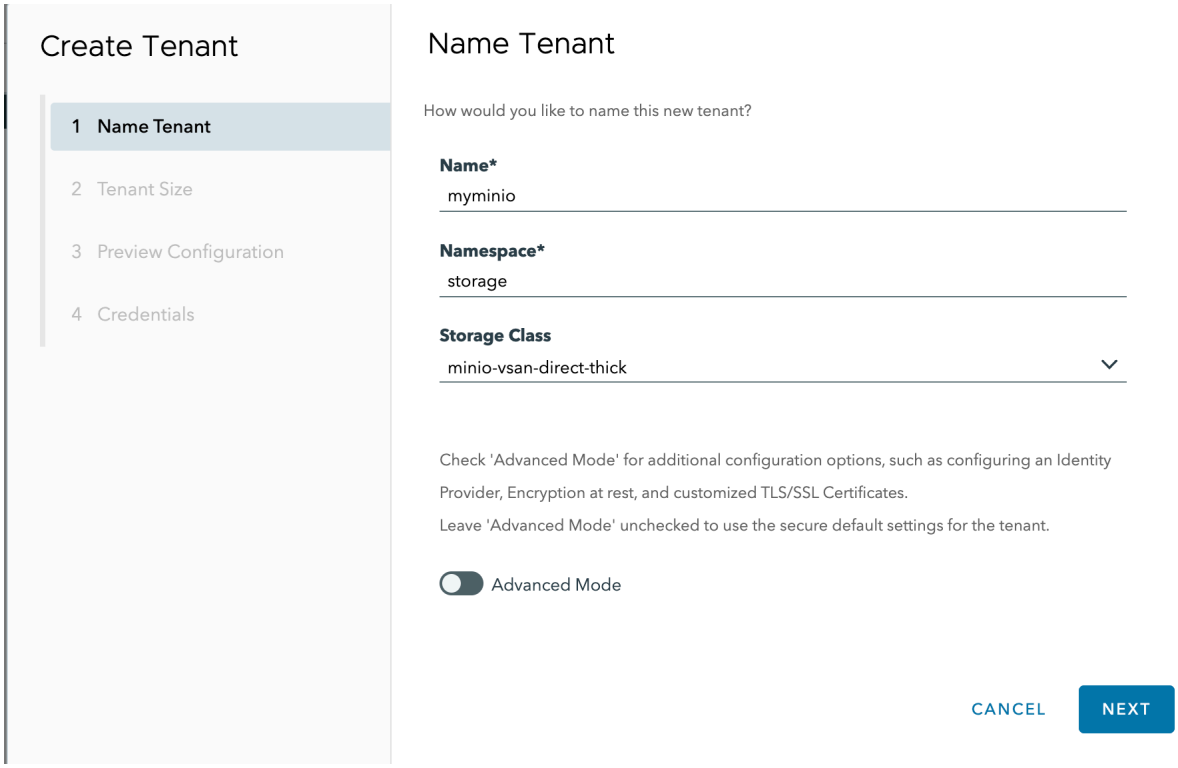
Create Tenant has four steps: **Name Tenant**, **Tenant Size**, **Preview Configuration**, **Credentials**. **Advanced Mode** adds steps for an identity provider, encryption at rest and custom TLS certificates; left off, the tenant gets the plug-in's secure defaults.

1. **Name Tenant**:
    - **Name**: the tenant name, e.g. myminio.
    - **Namespace**: a text field — type the vSphere Namespace prepared in step 4 (storage).
    - **Storage Class**: enabled once the namespace is entered; it offers the storage policies assigned to that namespace. Choose **minio-vsan-direct-thick**.

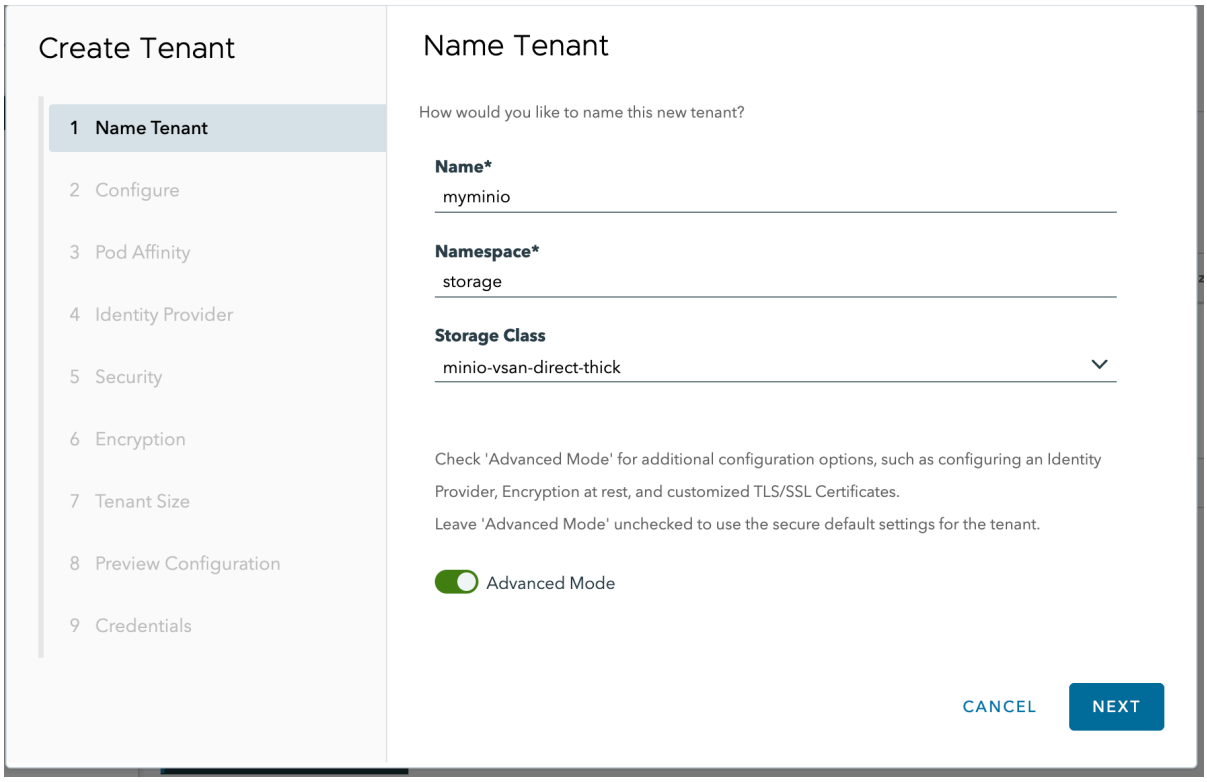
    NEXT stays disabled until Name and Namespace are filled.



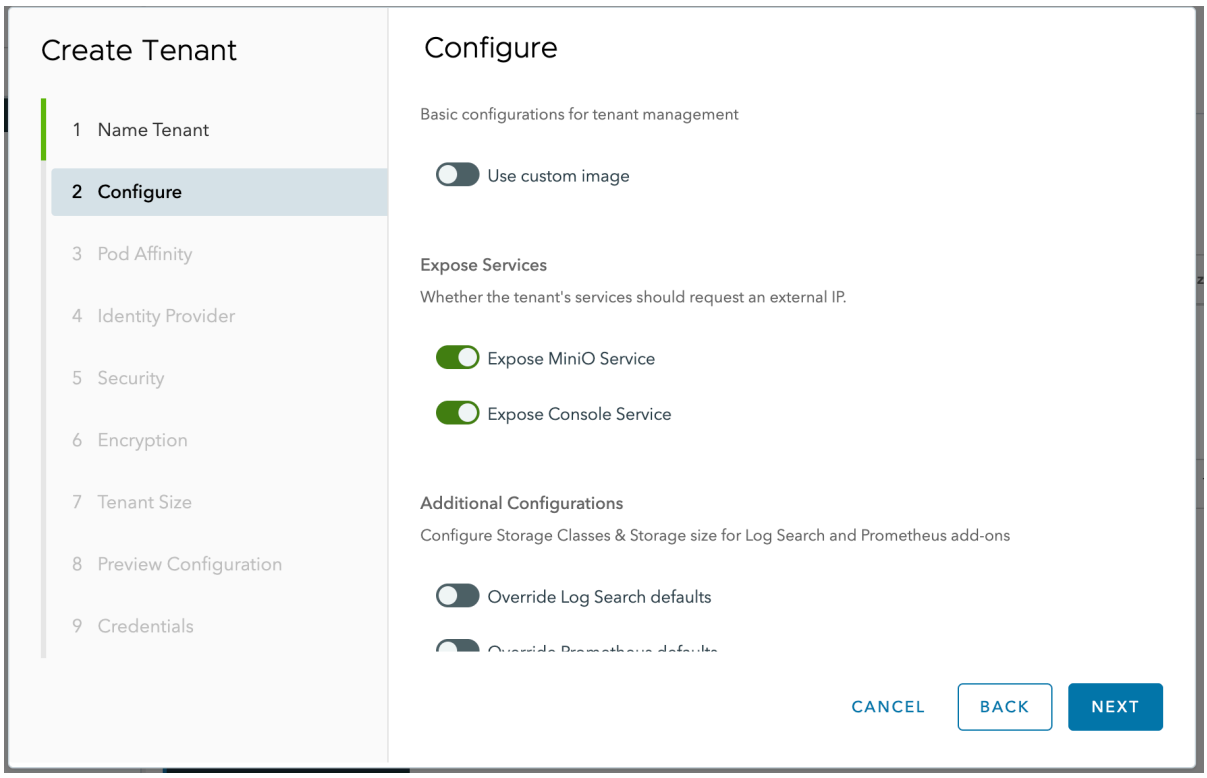
Filled in for the lab: myminio in storage, storage class minio-vsan-direct-thick:



With **Advanced Mode** on, the wizard grows from four to nine steps: **Name Tenant**, **Configure**, **Pod Affinity**, **Identity Provider**, **Security**, **Encryption**, **Tenant Size**, **Preview Configuration**, **Credentials**.

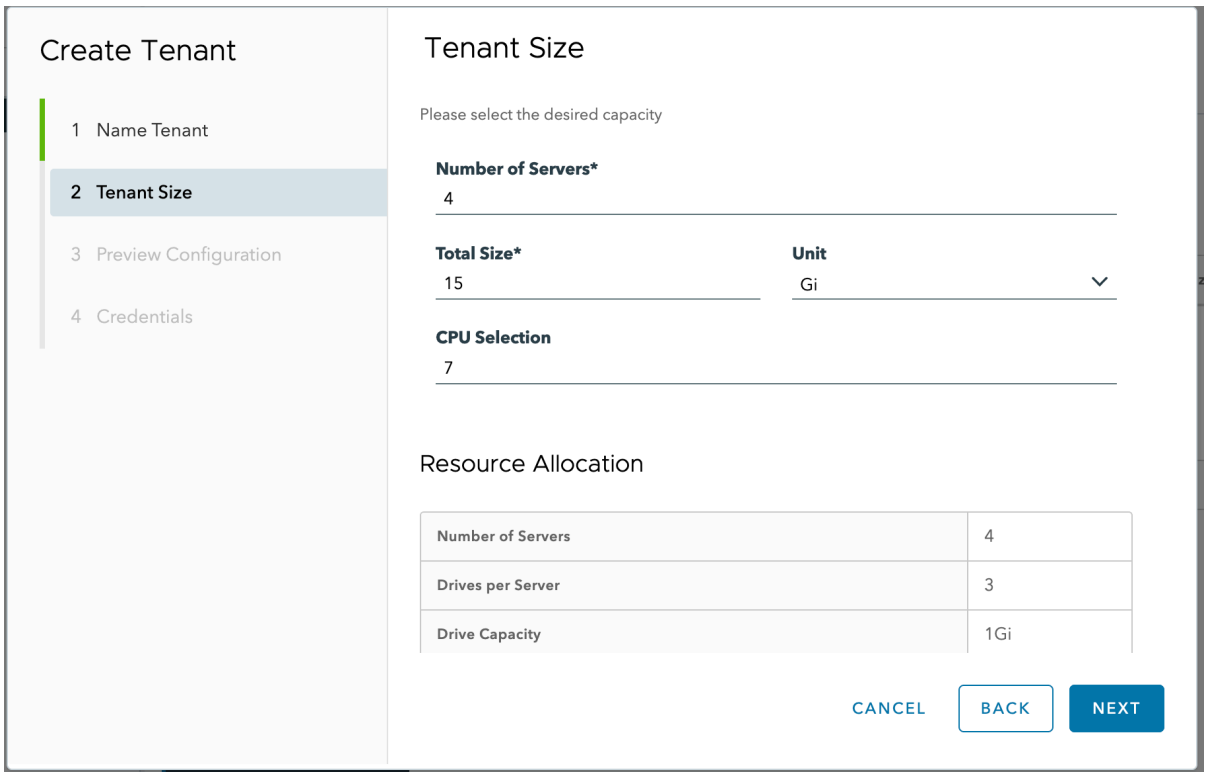


2. **Configure** (Advanced Mode): **Use custom image** (off), **Expose Services** — **Expose MinIO Service** and **Expose Console Service**, both **on** by default, so the tenant's services get an external address from the Supervisor's load balancer — and **Additional Configurations** to override the Log Search and Prometheus storage defaults. Basic mode uses the same defaults.

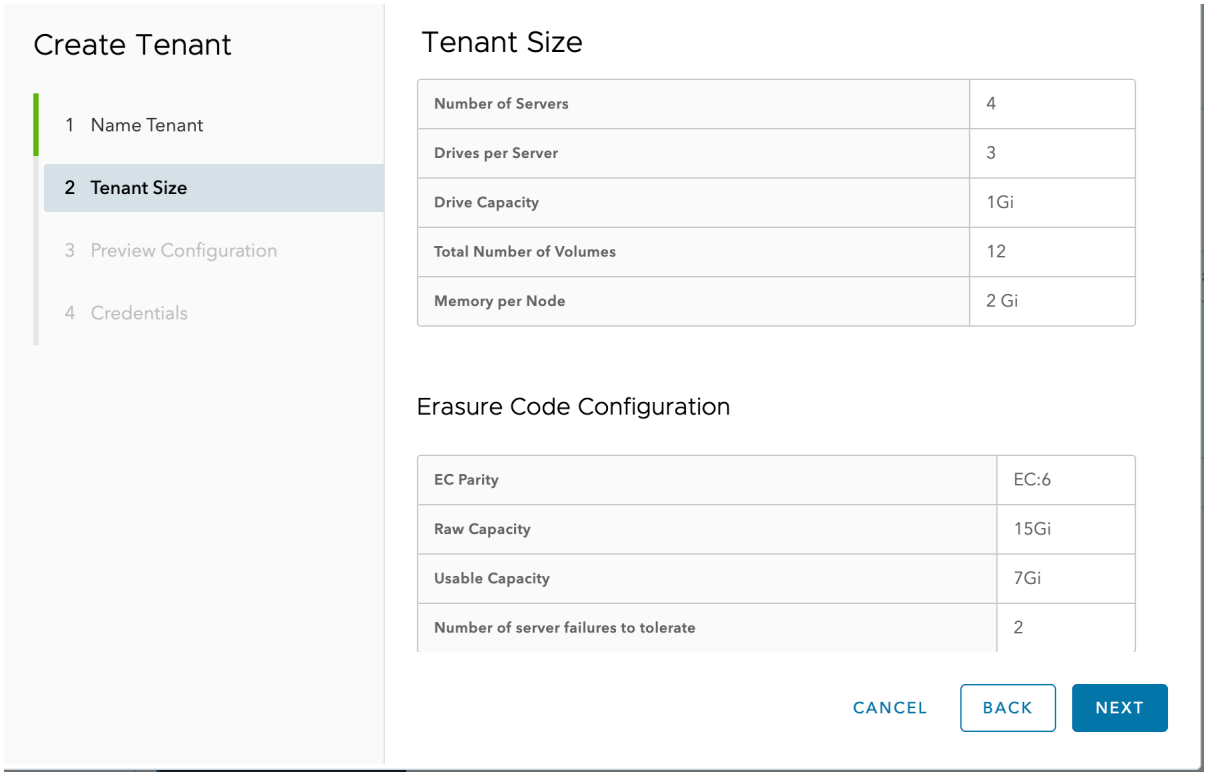


The MinIO image: MinIO no longer provides the open-source image 2.0.10 tenants use by default (Known issues). Turn **Use custom image** on and enter your copy in the service's registry project, e.g. registry.example.com/minio/minio:RELEASE.2024-02-09T21-25-16Z, and **leave Set/Update Image Registry off**. The plug-in gives a tenant created without **Set/Update Image Registry** a copy of the service's pull secret (<prefix>-regcred, with your registry's credentials) in its namespace, and the tenant pods pull with it: the MinIO image and the operator image they also run (the sidecar container and the validate-arguments init container). A registry set on the tenant replaces that secret, and the operator image then fails to pull.

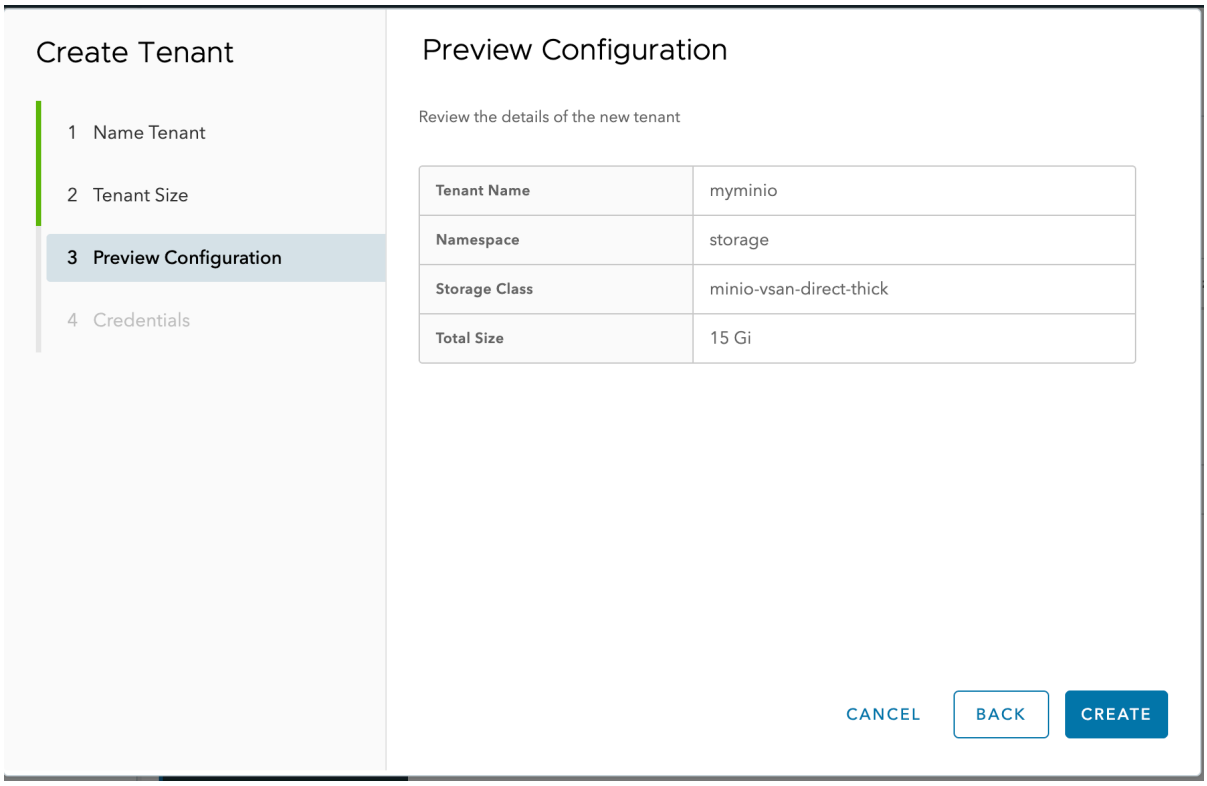
3. **Tenant Size** (basic mode, step 2): **Number of Servers** (at least 4), **Total Size** with its **Unit**, and **CPU Selection**. **Resource Allocation** shows how the plug-in spreads the capacity. With minio-vsan-direct-thick it uses **one drive per vSAN Direct disk on each host** — 3 here, so 4 servers × 3 drives — and sizes the drives from the smallest free disk. In the lab, 15 Gi gives 4 servers × 3 drives of 1 Gi.



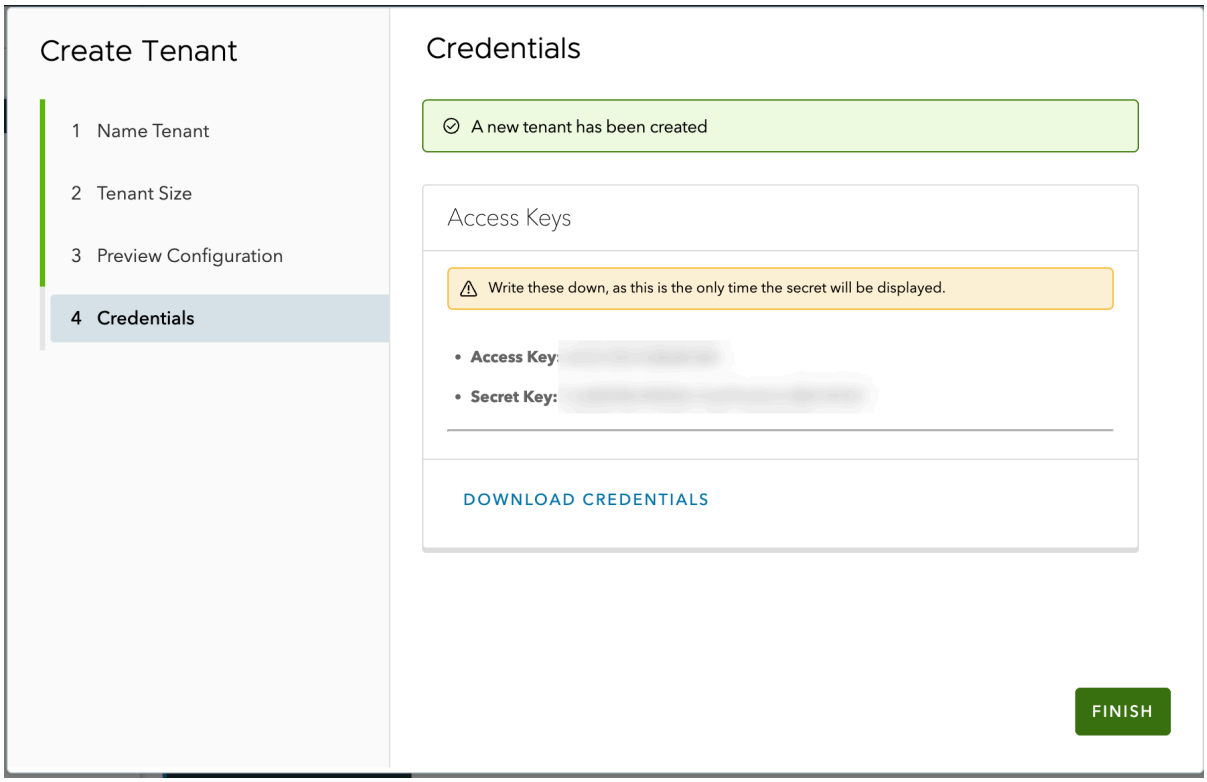
Further down the same step: **Total Number of Volumes** (12), **Memory per Node** (2 Gi), and the **Erasure Code Configuration** — **EC Parity** EC:6, **Raw Capacity** 15 Gi, **Usable Capacity** 7 Gi, **Number of server failures to tolerate** 2. Basic mode picks the parity itself; for 12 drives it chooses EC:6, half the drives.



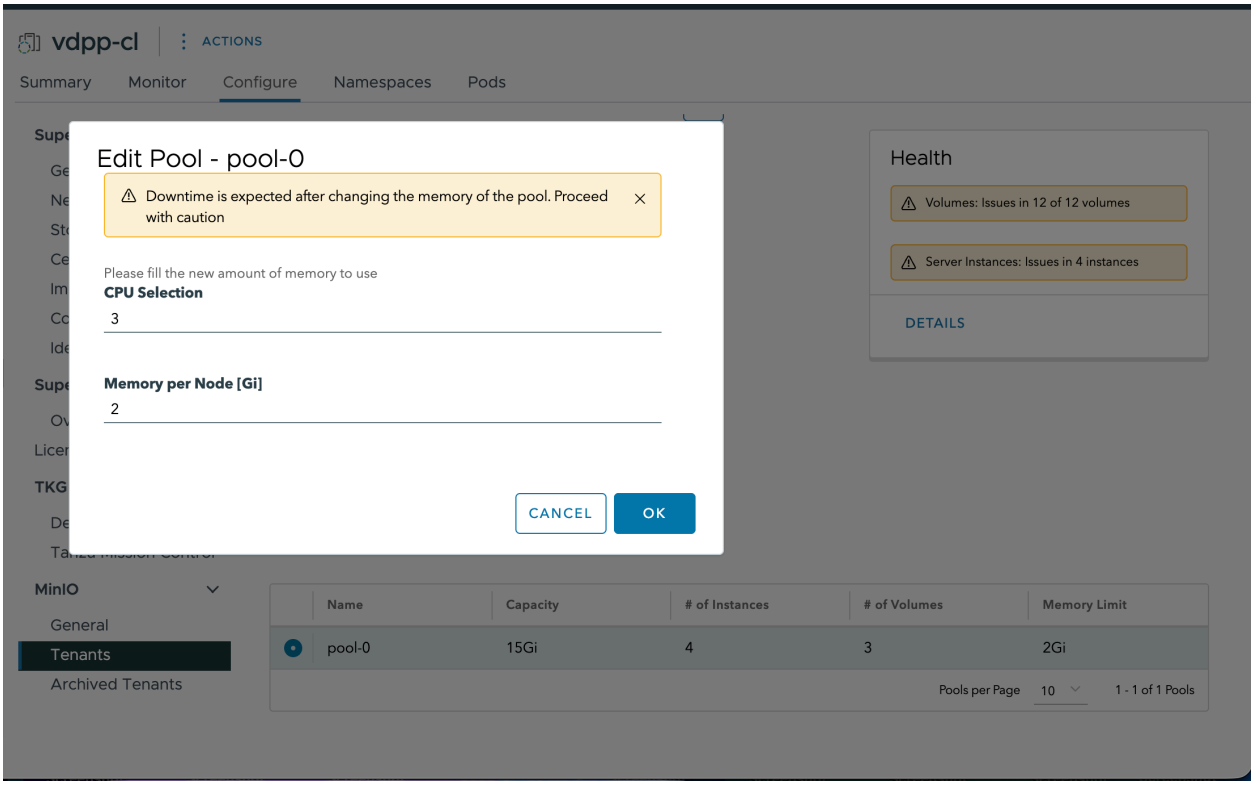
4. **Preview Configuration**: Tenant Name, Namespace, Storage Class and Total Size. Click **CREATE**.



5. **Credentials**: "A new tenant has been created", with the tenant's **Access Key** and **Secret Key**. They are shown only this once — note them or use **DOWNLOAD CREDENTIALS** — then **FINISH**.



CPU Selection defaults to a host's whole allocatable CPU (7 on the lab's 8-core hosts). vSphere Pods reserve what they request, so on small hosts the MinIO pods stay **Pending** with "DRS cannot find a host to power on or migrate the virtual machine". Lower it on Tenant Size, or afterwards from the tenant's pool: **Edit Pool** sets **CPU Selection** and **Memory per Node** ("Downtime is expected after changing the memory of the pool"); the pods are recreated one by one.

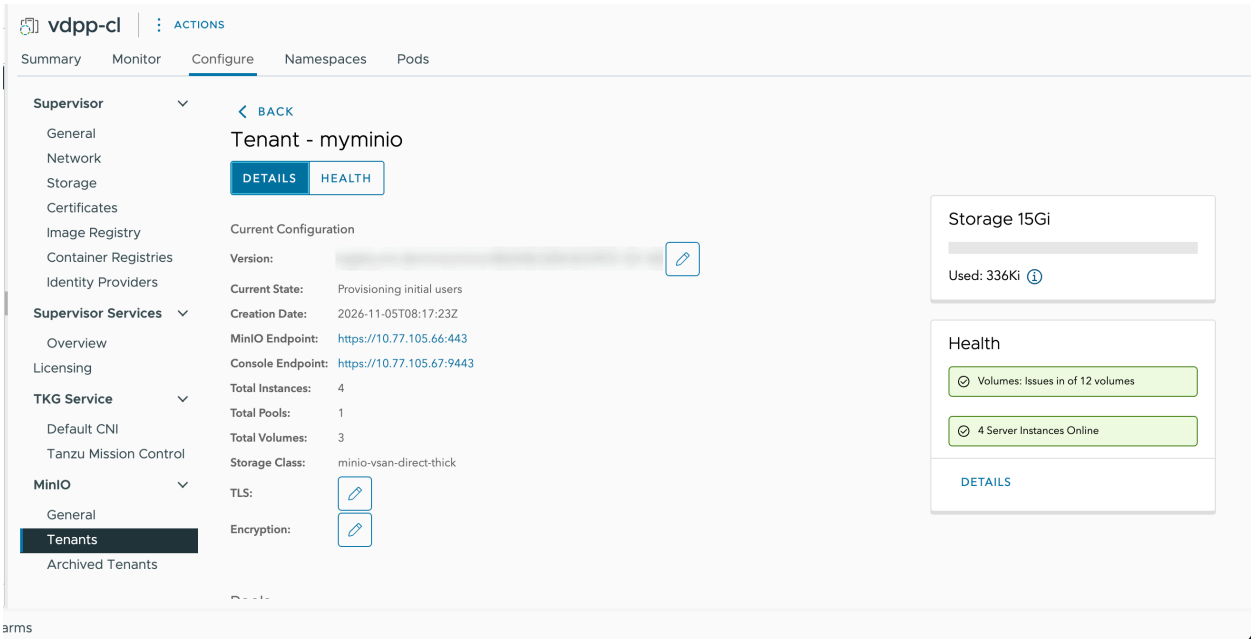


5.3 Check the tenant

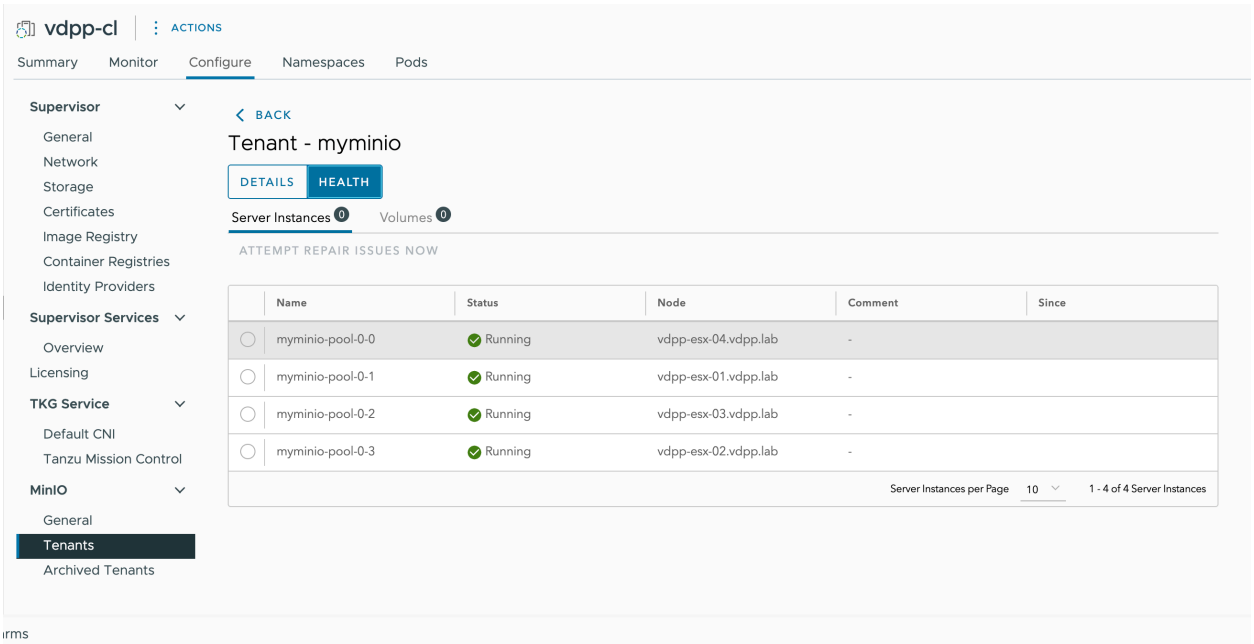
MinIO → Tenants → <tenant> opens the tenant page with **DETAILS** and **HEALTH**.

DETAILS shows the current configuration — image (**Version**), **Current State**, creation date, the **MinIO Endpoint** and **Console Endpoint**, instances, pools, volumes per server, storage class, and **TLS** / **Encryption** edit buttons — next to the **Storage** and **Health** cards.

The endpoints are listed in the vSphere Client on the tenant's **DETAILS** page (**Workload Management → Supervisors → <supervisor> → Configure → MinIO → Tenants → <tenant>**), as **MinIO Endpoint** (S3 API, port 443) and **Console Endpoint** (port 9443). They are external addresses that the Supervisor's load balancer gives the tenant's services. With kubectl, the same addresses are the EXTERNAL-IP of the tenant's services: kubectl get svc -n <namespace>.



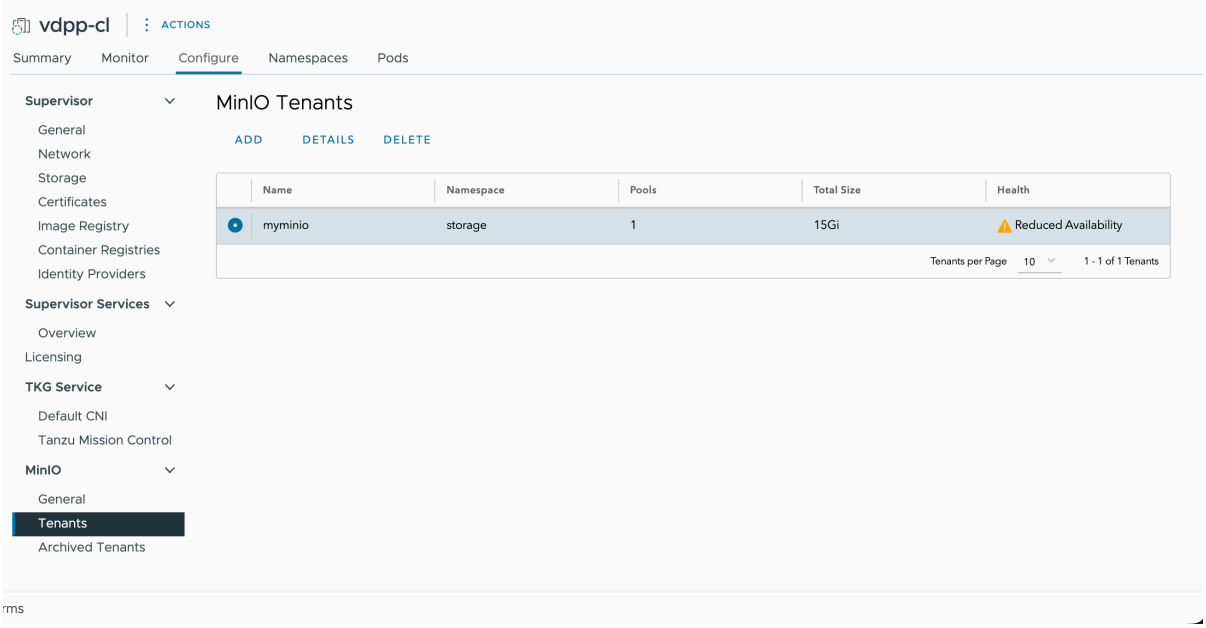
HEALTH lists the **Server Instances** and **Volumes** with their status, node and comments, and **ATTEMPT REPAIR ISSUES NOW**. A healthy tenant shows one Running instance per ESXi host:



From a workstation that reaches those addresses, the tenant answers on https://<minio-endpoint>/minio/health/live, and mc connects with the tenant's access and secret keys (--insecure for the self-signed certificate).

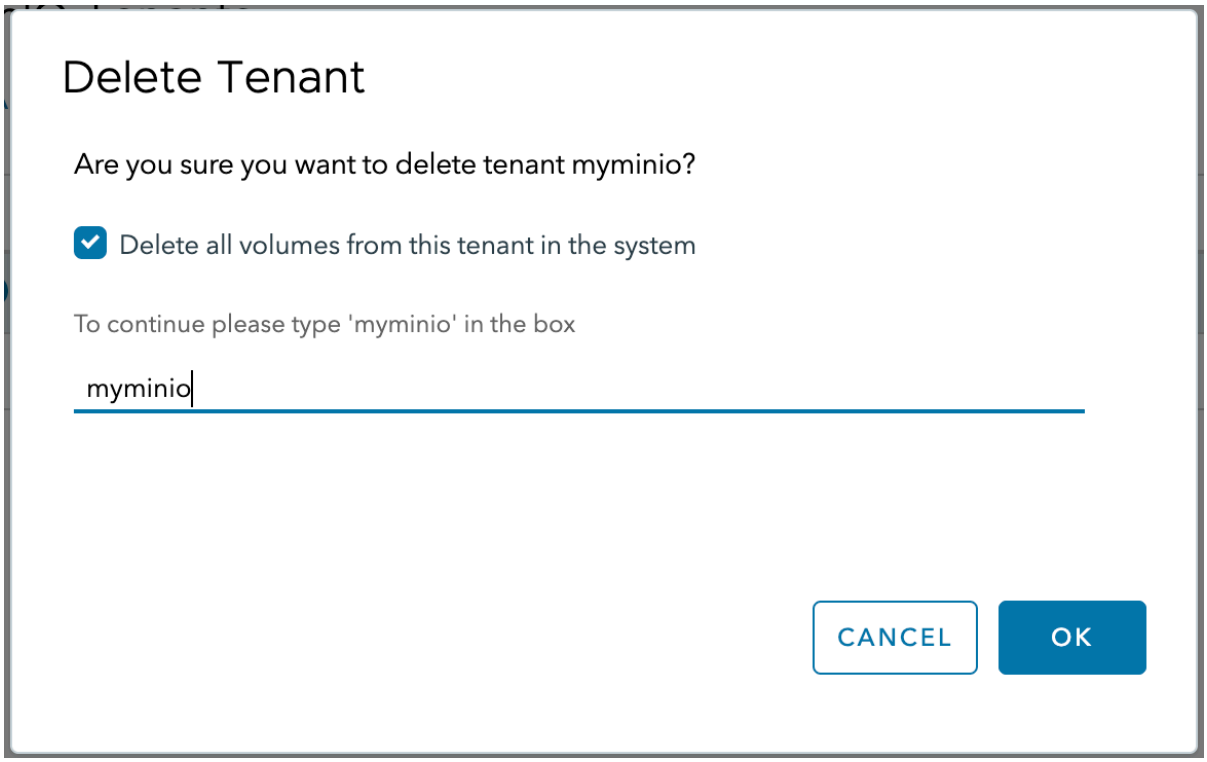
6. Delete a tenant

1. **MinIO → Tenants**: the list shows each tenant's namespace, pools, total size and health. Select the tenant and click **DELETE**.



2. **Delete Tenant** asks for confirmation:

   - **Delete all volumes from this tenant in the system**: ticked, the PVCs are deleted with the tenant and their vSAN Direct space is released; unticked, the volumes are kept and the tenant can be found under **Archived Tenants**.
   - Type the tenant name to enable **OK**.



With the volumes deleted, the tenant's pods, Services (and their external addresses), Secrets and PVCs are gone from the namespace within seconds, and the vSAN Direct storage pools are back to their previous free space. The vSphere Namespace itself and its storage policy remain.

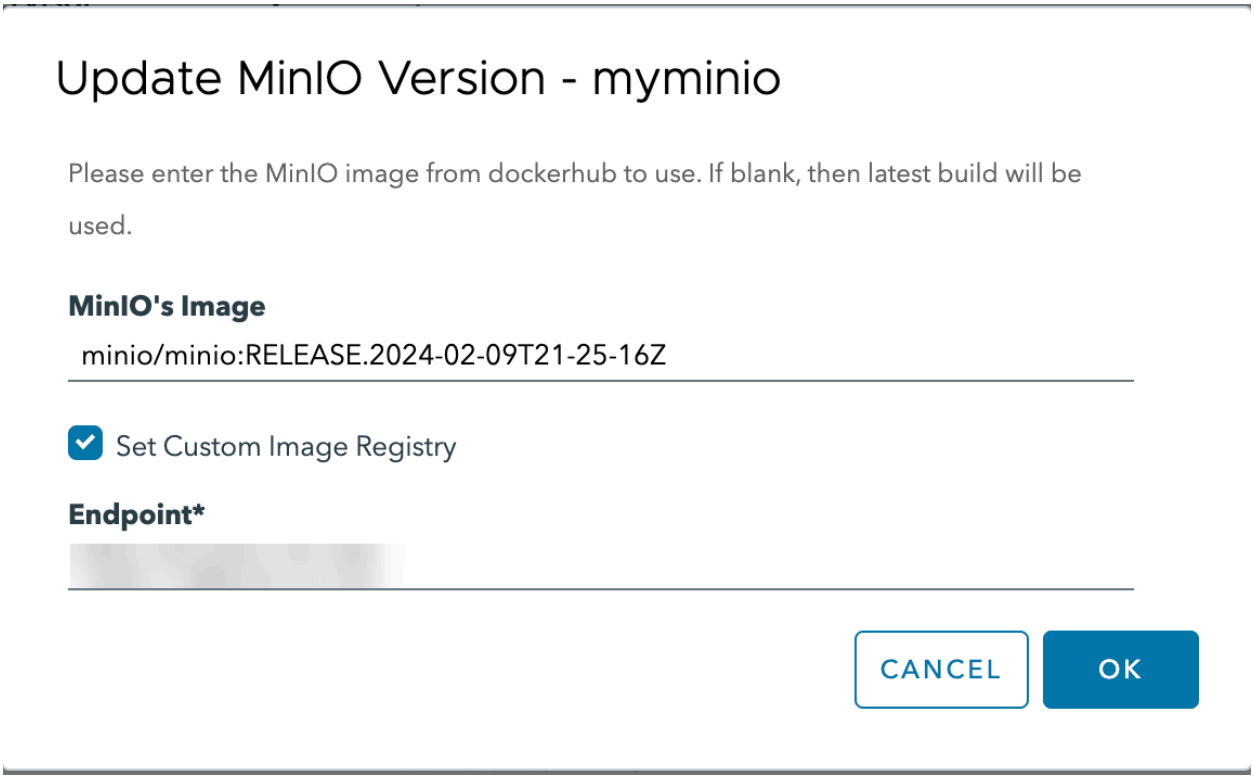
Known issues

The default MinIO image is retired. 2.0.10 creates tenants with the open-source image minio/minio:RELEASE.2024-02-09T21-25-16Z (the default of operator v5.0.16). The tag names no registry, so vSphere pulls it from Docker Hub. MinIO has retired the public open-source minio/minio images: neither Docker Hub nor Quay serves them any more (checked 2026-11-05). The MinIO pods of every new tenant therefore fail with ErrImagePull, the namespace events showing Failed to resolve image: … pull access denied, repository does not exist or may require authorization: server message: insufficient_scope, while the sidecar (minio/operator:v5.0.16) still pulls. The tenant stays in **Provisioning initial users** and its health shows issues in every volume and server instance.

MinIO does not distribute this image any more, and will not. To run a 2.0.10 tenant you need a copy of minio/minio:RELEASE.2024-02-09T21-25-16Z in a registry of your own, saved before the images were retired. Push it into the service's registry project (copy the images), and point the tenant at it from **MinIO → Tenants → <tenant> → DETAILS → Update MinIO**:

- **MinIO's Image**: registry.example.com/minio/minio:RELEASE.2024-02-09T21-25-16Z;
- **Set Custom Image Registry**: off, so that the tenant keeps the service's pull secret.

The screenshot below shows the earlier way, with **Set Custom Image Registry** on: do not use it while the service pulls from a private registry.



The same fields are under **Use custom image** in the wizard's Advanced Mode, for new tenants (5.2 Add the tenant, step 2).

A tenant with its own image registry cannot pull the operator image. A tenant takes a single pull secret: with **Set/Update Image Registry** (or **Set Custom Image Registry**) on, the plug-in gives it only that registry's credentials instead of the service's, and the tenant pods stay in ErrImagePull on the operator image. Keep the MinIO image in the service's registry project and leave that setting off.

"502 Bad Gateway" on MinIO → General. The internal TLS certificate of the plug-in expires, and the plug-in page in vCenter stops loading. Restart the plug-in deployment, wait for its pods to be Running, and reload the vSphere Client (Broadcom KB 449575):

```
kubectl rollout restart deployment -n svc-minio-domain-c<N> minio-vsphere
kubectl get pods -n svc-minio-domain-c<N>
```