# VMware Tanzu and Entrust CloudControl

## Integration Guide

2024-12-20

© Entrust Corporation. All rights reserved.

# Table of Contents


1. Introduction . . . 1
    1.1. Product configurations . . . 1
    1.2. Requirements . . . 1
2. Procedures . . . 2
    2.1. Download the CloudControl software . . . 2
    2.2. Deploy the CloudControl VM from the OVA . . . 2
    2.3. Power on the appliance . . . 3
    2.4. Configure the CloudControl virtual appliance . . . 3
    2.5. Set up the CloudControl GUI . . . 3
    2.6. VMware Tanzu prerequisites . . . 3
    2.7. VMware Tanzu setup . . . 3
    2.8. On-board the Tanzu clusters . . . 4
    2.9. View VMware Tanzu Kubernetes cluster inventory . . . 10
    2.10. Transfer root access control to CloudControl . . . 10
    2.11. Create a Trust Manifest Access Control Policy . . . 13
    2.12. Image registries . . . 16
    2.13. Image Deployment Control Policy . . . 18
    2.14. Configuration hardening . . . 29
3. Additional resources and related products . . . 37
    3.1. CloudControl . . . 37
    3.2. Entrust products . . . 37
    3.3. nShield product documentation . . . 37

# Chapter 1. Introduction

This guide describes how to integrate VMware Tanzu Kubernetes clusters with Entrust CloudControl. Entrust CloudControl organizes a cluster inventory into categories relating to the Tanzu deployment. Entrust CloudControl uses role and asset-based access control to help the user define who can do what to which cluster objects. It also uses image deployment control policies that can be applied to a cluster infrastructure. This ensures ongoing compliance with your organization security policies.

## 1.1. Product configurations

Entrust has successfully tested the integration of Entrust CloudControl with VMware Tanzu in the following configurations:

| System | Version |
|---|---|
| VMware vCenter | 7.0.1 U1 (build-16858589) |
| Kubernetes Version | v1.18.19+vmware.1 |
| Entrust CloudControl | 6.6.0 |

## 1.2. Requirements

Before starting the integration process, familiarize yourself with:

- The documentation and setup process for VMware Tanzu.
- The documentation and setup process for Entrust CloudControl. The online documentation contains everything needed to successfully install and deploy CloudControl.

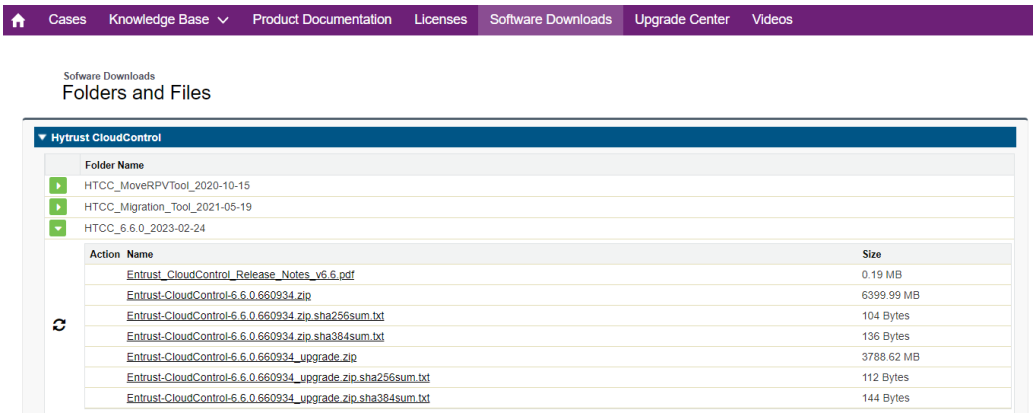
Entrust recommends that you allow only unprivileged connections unless you are performing administrative tasks.

# Chapter 2. Procedures

This guide uses a standalone CloudControl deployment and does not use Active Directory. All users are local to the system. CloudControl supports a cluster environment. For more information refer to the Entrust CloudControl Installation Guide in the online documentation.

## 2.1. Download the CloudControl software

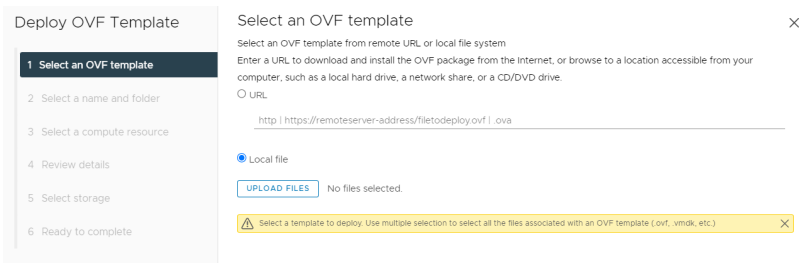
1. Go to https://my.hytrust.com/s/software-downloads.
2. Log in and select **HyTrust CloudControl**.
3. Open the folder `HTCC_6.6.0_2023-02-241`. This folder contains version 6.5.0 that was used in this guide.
4. Select the `Entrust-CloudControl-6.6.0.660934.zip` link to download the file.




5. After the file has been downloaded, open the ZIP file to access to the OVA file.

## 2.2. Deploy the CloudControl VM from the OVA

1. Log in to vCenter.
2. Select the cluster to create the CloudControl VM in.
3. From the **Actions** menu, select **Deploy OVF template**.



4. Select **Local File** and upload the CloudControl OVA file.
5. Select **Next**

Follow the instructions during the deployment as needed.

For more information refer to Installing CloudControl from an OVA in the online documentation.

## 2.3. Power on the appliance

1. Log in to the vSphere Client.
2. Locate the Entrust CloudControl virtual machine in the inventory.
3. Right-click the CloudControl virtual machine and select **Power > Power On**.

## 2.4. Configure the CloudControl virtual appliance

This guide uses a Standalone Node setup. For more information refer to Creating a Standalone Node in the online documentation.

## 2.5. Set up the CloudControl GUI

After the standalone node has been configured, finish the setup using the GUI. For more information refer to Setting Up the CloudControl GUI in the online documentation.

## 2.6. VMware Tanzu prerequisites

CloudControl has some prerequisites for VMware Tanzu that must be put in place.

For more information refer to VMware Tanzu Prerequisites in the online documentation.

## 2.7. VMware Tanzu setup

This guide does not describe the deployment of a VMware Tanzu cluster. Refer to the VMware documentation for details.

When the VMware Tanzu cluster is set up, use the cluster kubeconfig file to onboard the cluster into CloudControl. A client machine with `kubectl` installed is required to run commands specified in this guide.

The examples in this guide use a CentOS 8 virtual machine which will be referred to as the kubectl node from now on.

1. Log in to the kubectl node client machine.
2. On the kubectl node log into the VMware Tanzu Management/Supervisor cluster:

```
% kubectl vsphere login --vsphere-username administrator@vsphere.local --server=https://xx.xxx.xxx.xx
--insecure-skip-tls-verify
```

   When this command has executed successfully, use the `$HOME/.kube/config` file for onboarding. This will be the config file to onboard the Management/Supervisor cluster. Save this file in the `$HOME` directory and name it `management.kubeconfig.txt`.

```
% cp $HOME/.kube/config $HOME/management.kubeconfig.txt
```

3. To get the Tanzu Kubernetes cluster config file, log into the Tanzu Kubernetes cluster:

```
% kubectl vsphere login --vsphere-username administrator@vsphere.local --server=https://xx.xxx.xxx.xx
--insecure-skip-tls-verify --tanzu-kubernetes-cluster-namespace qanamespace --tanzu-kubernetes-cluster-name
qa-tkg-cluster-01
```

   In this example, the namespace for our cluster is **qanamespace** and the cluster name is **qa-tkg-cluster-01**. These are just examples. Enter the names that apply to the environment in use.

   When this command has executed successfully, use the `$HOME/.kube/config` file for on-boarding. This will be the config file to onboard the Tanzu Kubernetes cluster. Save this file the `$HOME` directory and name it `tanzucluster.kubeconfig.txt`.

```
% cp $HOME/.kube/config $HOME/tanzucluster.kubeconfig.txt
```

4. `$HOME/.kube/config` is the file needed to onboard the clusters in CloudControl. For Tanzu, use the following files:
   a. `management.kubeconfig.txt` - for the Management/Supervisor cluster.
   b. `tanzucluster.kubeconfig.txt` - for the Tanzu Kubernetes cluster.

Both clusters will have to be on-boarded into CloudControl.

Now add the Tanzu clusters into CloudControl, see On-board the Tanzu clusters.

## 2.8. On-board the Tanzu clusters

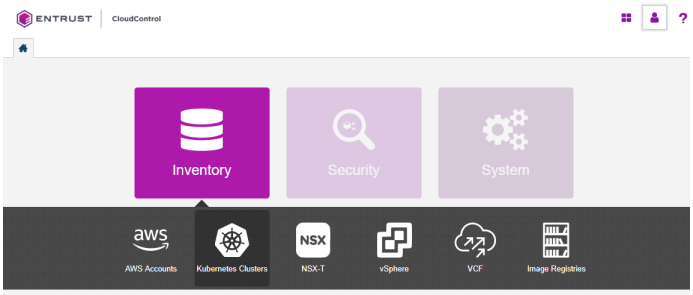
On-boarding is the process of adding the Tanzu Kubernetes clusters into CloudControl.

For more information refer to Adding a VMware Tanzu Cluster in the online documentation.

For Tanzu, import the Tanzu Management cluster and at least a Tanzu Kubernetes cluster.

### 2.8.1. On-boarding the Tanzu Management cluster

1. From the **Home** tab, select **Inventory > Kubernetes Clusters**.




2. On the **Clusters** page, select **Actions > Add Kubernetes Cluster**.

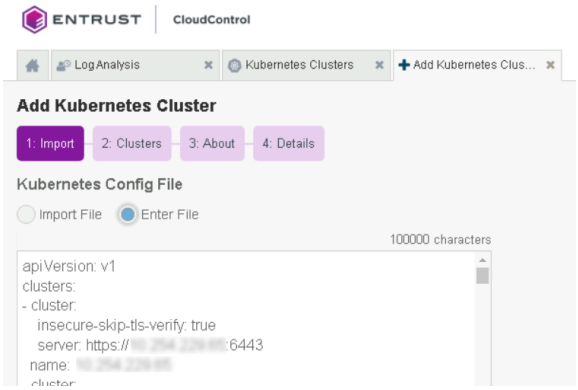
   If there are no clusters in the system, **Add Kubernetes Cluster** is also available on the **Kubernetes Clusters** page.

3. On the **Add Kubernetes Cluster Import** tab, choose one of the following:

   - Select **Import File**, then select **Browse** and choose the kubeconfig file to import.
   - Select **Enter Text**, then paste the contents of the kubeconfig file as plain text.

   A kubeconfig file is a configuration file written in YAML that describes the cluster.

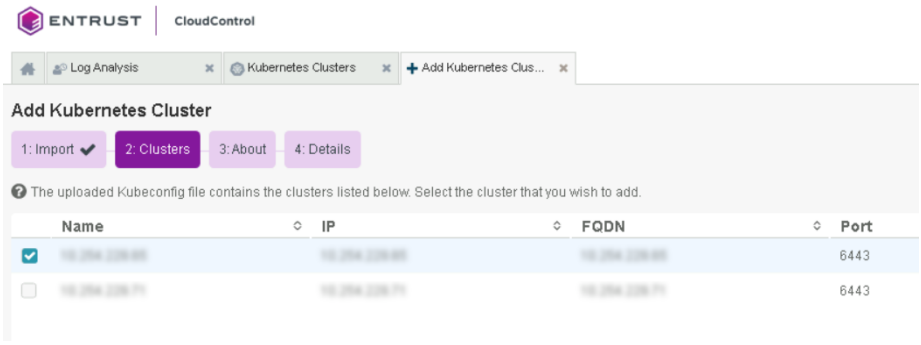
   To import the Management cluster, use the Management cluster config file produced earlier, `management.kubeconfig.txt`.



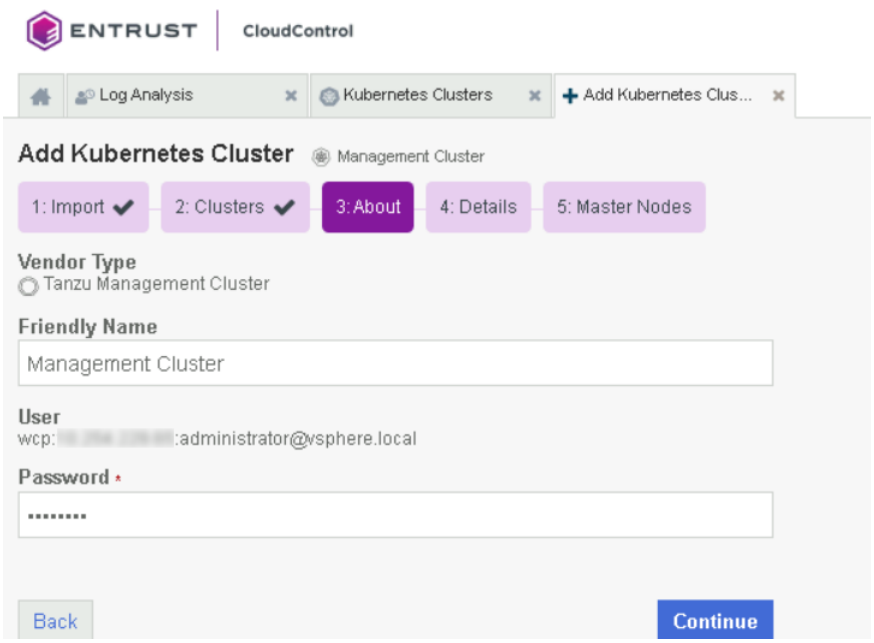

4. Select **Enter Text**, then paste the contents on the Management cluster config file as plain text.

5. Select **Continue**.
6. On the **Clusters** tab, select the Tanzu Management cluster to add and select **Continue**.

   Only one cluster can be selected. Select the Tanzu Management cluster IP address or FQDN.

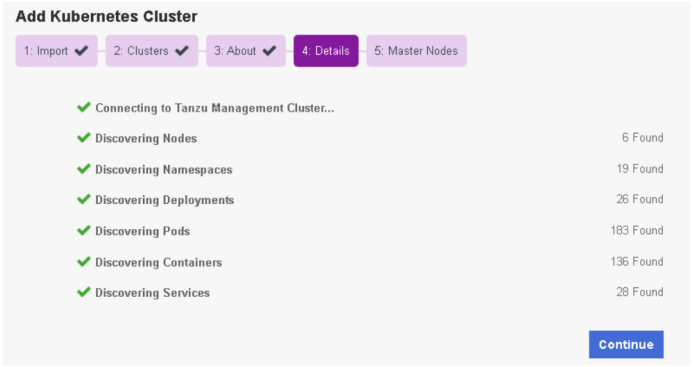



7. On the **About** tab, enter the following:
   - **Friendly Name**: Enter the user-facing name for the cluster.
   - **Password**: Enter the password for the user selected who has access to the cluster. In this example, that is the same password used by `administrator@vsphere.local`.

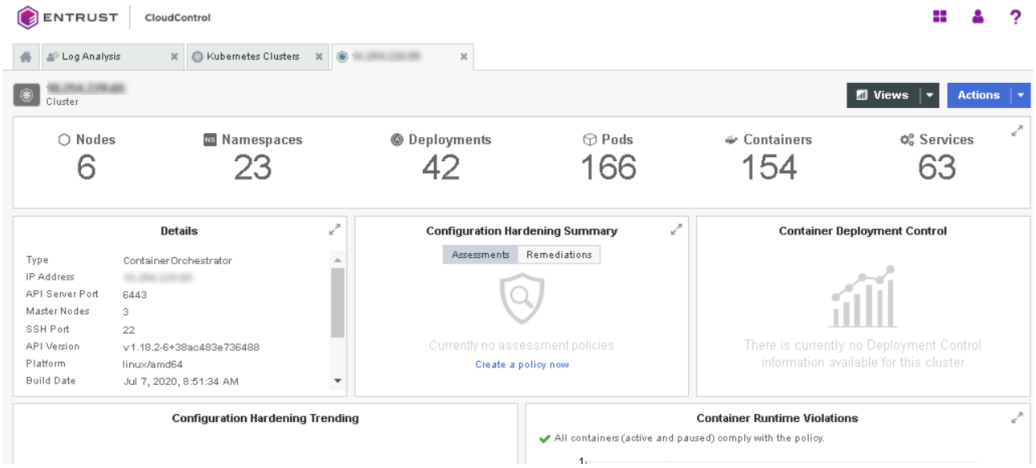



   For the **Vendor Type**, CloudControl detects that this is a Tanzu Management cluster.

8. Select **Continue**.
9. On the **Details** tab, monitor the process.



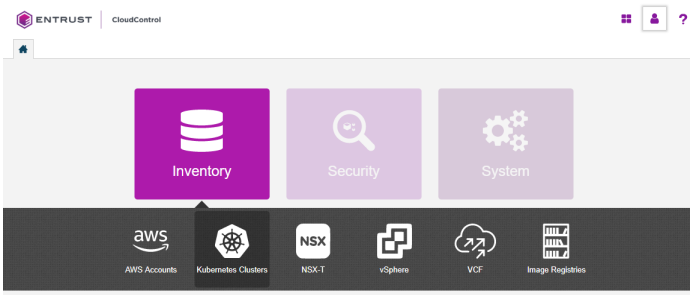

10. Select **Continue**.
11. On the **Master Nodes** tab, a list of nodes will be displayed with their name, IP, and hostname information.
12. Select **Continue**.
13. After the cluster has been imported into CloudControl the cluster dashboard is shown.




## 2.8.2. On-boarding the Tanzu Kubernetes cluster

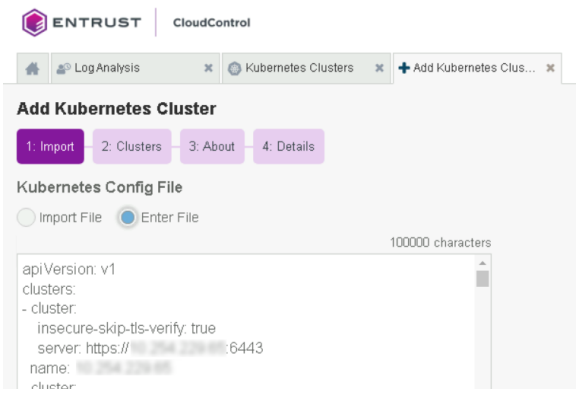
After the Tanzu Management cluster has been on-boarded, import a Tanzu Kubernetes cluster.

1. From the **Home** tab, select **Inventory > Kubernetes Clusters**.



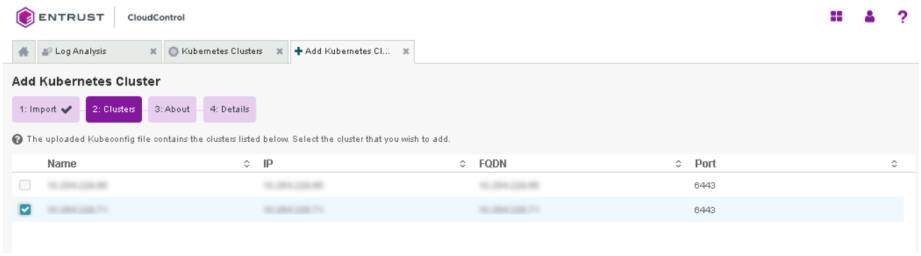
2. On the **Clusters** page, select **Actions > Add Kubernetes Cluster**.
3. On the **Add Kubernetes Cluster Import** tab, select **Enter Text**, then paste the contents of the kubeconfig file as plain text.

   To import the Tanzu Kubernetes cluster, use the Tanzu cluster config file `tanzucluster.kubeconfig.txt`.

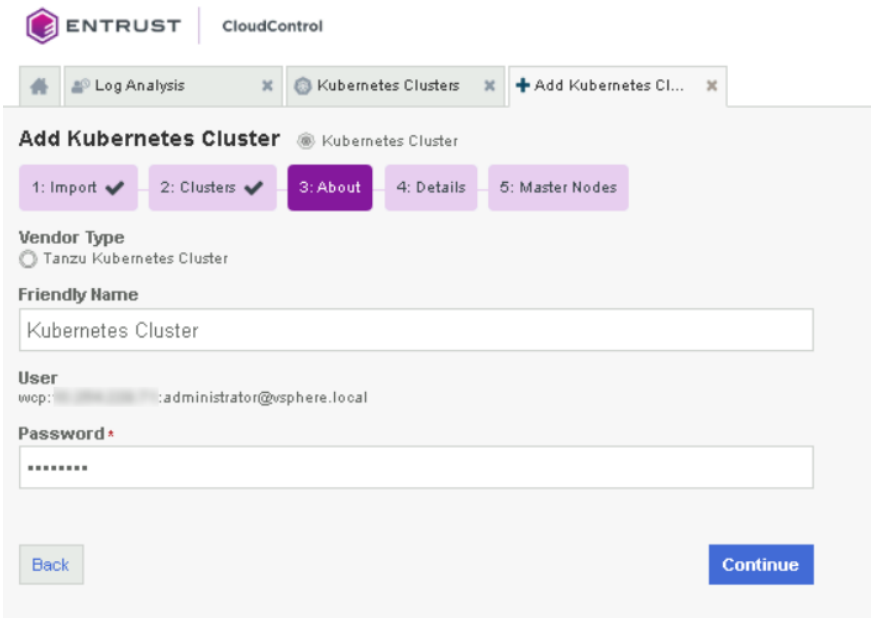



4. Select **Enter Text**, then paste the contents on the Tanzu Kubernetes cluster config file as plain text.
5. Select **Continue**.
6. On the **Clusters** tab, select the Tanzu Kubernetes cluster to add and select **Continue**.

   Only one cluster can be selected. Select the Tanzu Kubernetes cluster IP address or FQDN.

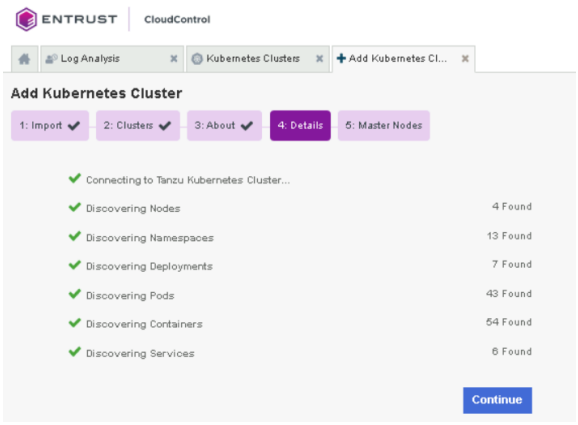



7. On the **About** tab, enter the following:
   - **Friendly Name**: Enter the user-facing name for the cluster.
   - **Password**: Enter the password for the user who has access to the cluster. In this example, that is the same password used by `administrator@vsphere.local`.




For the **Vendor Type**, CloudControl detects that this is a Tanzu Kubernetes cluster.

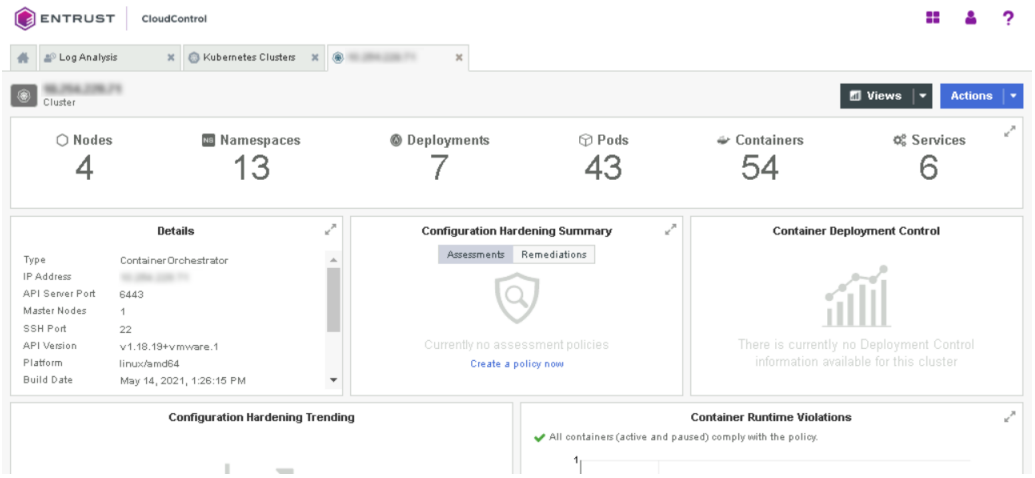
8. Select **Continue**.
9. On the **Details** tab, monitor the process.




10. Select **Continue**.
11. On the **Master Nodes** tab, a list of nodes will be displayed with their name, IP, and hostname information.
12. Select **Continue**.
13. In the **Enable Access Control** dialog, either:
    a. Select **Enable Access Control** to enable the ROOT Access Control Trust Manifest. CloudControl will take control of managing access to the cluster. Create a Trust Manifest Access Control Policy to be able to create and manage objects in the cluster.

b. Select **No, not now** to wait until later. This feature can be enabled after the cluster has been onboarded.

14. After the cluster has been imported into CloudControl the cluster dashboard is shown.




## 2.9. View VMware Tanzu Kubernetes cluster inventory

From the **Home** page, select **Inventory > Kubernetes Clusters** to view the **Kubernetes Clusters** page. From here, in depth information of all of the objects in that cluster can be viewed, as well as any tags or policies related to those objects. This information is included in a dashboard or a resource page.

For more information refer to Viewing Kubernetes Inventory in the online documentation.

Also refer to Navigating Kubernetes Inventory View Pages in the online documentation.

## 2.10. Transfer root access control to CloudControl

When root access control is enabled in the Kubernetes cluster, access control of the Tanzu Kubernetes cluster is transferred to CloudControl. This means that management of objects in the cluster will be controlled by CloudControl rules. These rules must be created and defined in CloudControl for the user to be able to create, edit, and delete objects in the cluster.

### 2.10.1. Before root access control is enabled

During the on-boarding process, root access was not enabled for the imported cluster. As a result, the user can still create objects at the Tanzu level.

For example, the user could create a pod. The `pod.yaml` file is below:

```
apiVersion: v1
kind: Pod
metadata:
    name: tutum-centos3
spec:
    containers:
    - name: tutum-ssh-server3
      image: tutum/centos
```

To create the pod:

```
% kubectl vsphere login --vsphere-username administrator@vsphere.local --server=https://xx.xxx.xxx.xx \
  --insecure-skip-tls-verify --tanzu-kubernetes-cluster-namespace qanamespace \
  --tanzu-kubernetes-cluster-name qa-tkg-cluster-01

Password:
Logged in successfully.

You have access to the following contexts:
   xx.xxx.xxx.xxx
   ameynamespace
   ameynamespace-xx.xxx.xxx.xxx
   amolnamespace
   qa-tkg-cluster-01
   qanamespace
   qanamespace-xx.xxx.xxx.xxx
   saketnamespace
   saketnamespace-xx.xxx.xxx.xxx
   subbanamespace
   wld1

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>`

% kubectl create -f pod.yaml

pod/tutum-centos3 created
```

The pod is created successfully. This process will fail if root access control is enabled using default HyTrust Global Access Control Policy, which denies all operations. See Enable root access control.

To delete the pod:

```
% kubectl get pods
NAME            READY   STATUS    RESTARTS   AGE
tutum-centos3   1/1     Running   0          3m57s

% kubectl delete pod tutum-centos3
pod "tutum-centos3" deleted
```

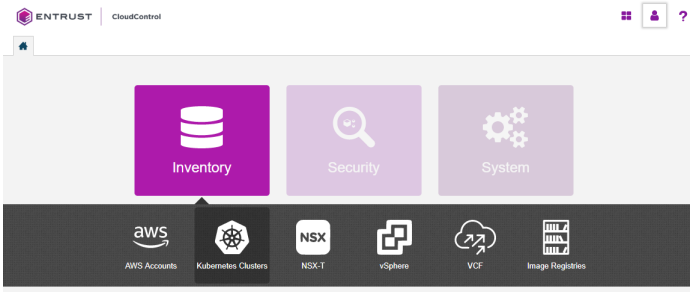
The pod is deleted successfully. This process will fail if root access control is enabled using default HyTrust Global Access Control Policy, which denies all operations. See Enable root access control.

## 2.10.2. Enable root access control

When root access control is enabled for the Kubernetes cluster, access control of the Tanzu Kubernetes cluster is transferred to CloudControl.

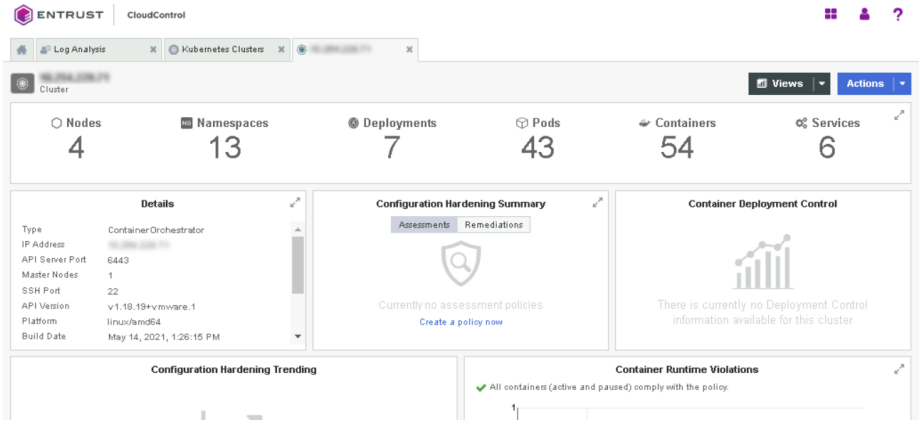
To enable root access control:

1. From the **Home** tab, select **Inventory > Kubernetes Clusters**.



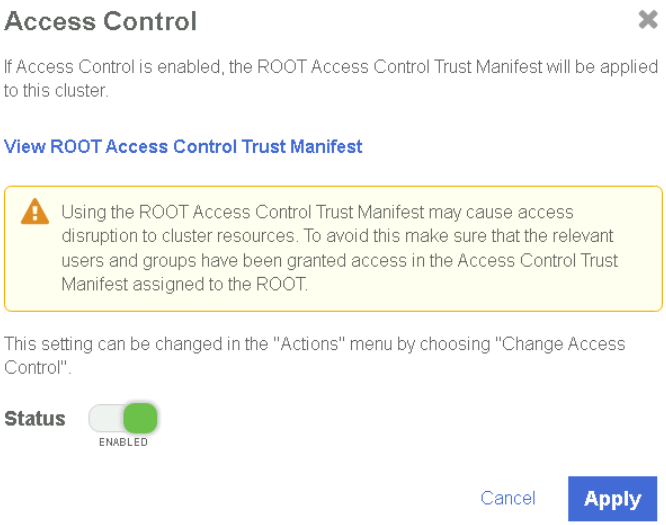

2. Select **Management** to view the clusters.
3. Select the Tanzu Kubernetes **Cluster Name** link.

   This action will display the details about the selected cluster.




   The **Details** section of the page shows that **Access Control** is **Disabled**.

4. Select the **Disable** link to **Enable Root Access Control**.




5. In the **Access Control** dialog, set **Status** to **Enabled** and select **Apply**.

After root access is enabled, it is not possible to work with objects directly. For example, the user could create a pod:

```
% kubectl create -f pod.yaml

Error from server (Forbidden): error when creating "pod.yaml": admission webhook "accesscontrol.hytrust.com"
denied the request: Permission Denied by HyTrust Security Policy.
```

In this example, it is not possible to create the pod, as CloudControl now controls permissions on the cluster. To perform these actions, user permissions must be granted through a policy, see Create a Trust Manifest Access Control Policy.

## 2.11. Create a Trust Manifest Access Control Policy

After root access control is enabled in the cluster, the user is unable to create objects in the cluster. This is because cluster permissions are then managed by CloudControl.

This section describes the process to create a Trust Manifest Access Control policy, so a user can manage objects in the cluster.

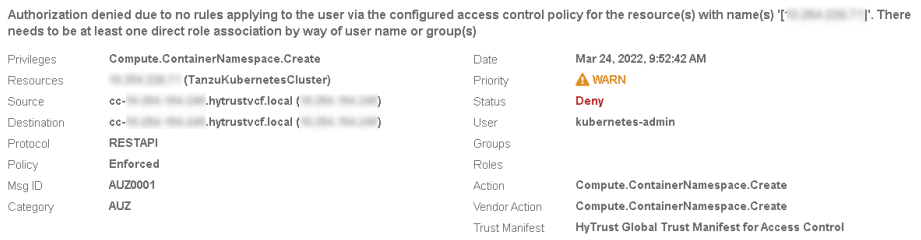
### 2.11.1. Log analysis

Use the logs in the system to check why access has been denied for a request. From this, create a Trust Manifest Access Control Policy to allow the user to perform the request successfully. For more information refer to Log Analysis in the online documentation.

1. Go to the **Home** tab, select **Security > Log Analysis**.

2. Select the record that shows the **Deny** status to see the reason for the denial.

   Look for an entry similar to the following:




   In this example, a request to create a pod was denied.

Now create a Trust Manifest Access Control Policy to allow the user to create the pod.

## 2.11.2. Create the Access Control Policy

This section describes how to use an Access Control Policy to allow users to create pods in the Tanzu cluster.

In the example below, the Access Control Policy will allow the `kubernetes:admin` user to create pods in the cluster. Any other user in the system will be denied access.

For details of how to create an Access Control Policy, refer to Creating an Access Control Trust Manifest from the CloudControl GUI in the online documentation.

1. From the **Home** tab, select **Security > Trust Manifests**.
2. On the **Manage Trust Manifests** page, select **Create Trust Manifest**. This is the plus sign in the GUI.
3. On the **Details** tab of the **Create Trust Manifest** page, enter the **Name** and optional **Description** for the Trust Manifest.
4. For **Policy Type**, select **Access Control**.
5. In the **Access Control Policy** section, enter the name of the rule. That is, **ACP**.
6. For **Rule Type**, select **Allow**.
7. For **Role**, select **ASC_ContainerInfraAdmin**.

   This is the role that controls all operation related to creating objects in cluster.

8. Under **Subjects**:
   a. For **Type**, select **Kubernetes**.
   b. For **Group or User**, add:

      - **k8s::user:kubernetes:admin**

- **k8s::user:sso:administrator@vsphere.local**

9. Select **Validate** to validate the policy.
10. Select **Save** to save the policy.
11. Select **Publish** to publish the policy.

    When the policy is published, it prompts the user to assign resources to the policy. Select the Tanzu Kubernetes cluster and select **Assign**.

12. Select **Close**.

### 2.11.3. Test the Access Control Policy

To test if the Access Control policy is working, create the pod:

```
% kubectl create -f pod.yaml

pod/tutum-centos3 created
```

The request is successful.

To test the deletion of a pod:

```
% kubectl delete pod tutum-centos3

Error from server (Forbidden): admission webhook "accesscontrol.hytrust.com" denied the request: Permission
Denied by HyTrust Security Policy, Resource not found.
```

In this example, the resource is not found in CloudControl. This is because the CloudControl has not updated its cluster inventory yet. When a cluster is onboarded, it creates an internal job that runs every 5 minutes to update the cluster inventory. This means that the only way to delete the pod in this case is to:

- Wait for the cluster inventory job to complete.
- Manually sync the inventory.

To manually sync the inventory:

1. In the **Cluster Detail** tab, select **Actions > Sync Inventory**.
2. Select **Initiate Sync**.

After the sync completes, delete the pod. For example:

```
% kubectl delete pod tutum-centos3
```

```
pod "tutum-centos3" deleted
```

The pod deletes successfully.

## 2.12. Image registries

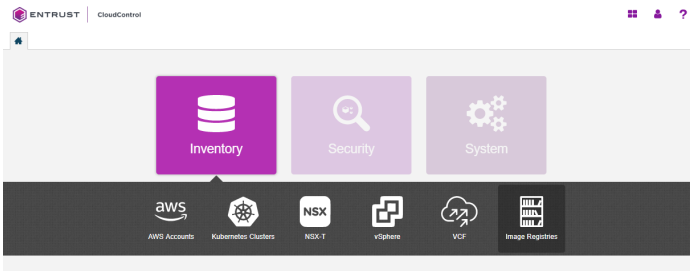
An image registry is a service that stores repositories and images. Each repository contains one or more version of the same image. All images in a repository must have the same name, and be differentiated by tags. The tag name corresponds to the version of the image. The most recent image is also tagged as 'latest'.

Image registries are not protected by CloudControl, but adding a registry allows CloudControl to discover valuable information about the registry, such as the number of images and their specific vulnerabilities.

This allows more detailed rules when creating an Image Deployment Control Policy which will be discussed later in this guide. Entrust strongly suggests adding private image registries to CloudControl for better control during image deployment in Kubernetes.

Add the private registry to CloudControl: xxxx.yyyy.zzz

1. In the **Home** tab, Select **Inventory > Image Registries** to view the registries that have been added to CloudControl.




2. On the **Image Registries** page, select **Actions > Add Registry**.

   If there are no registries in the system, select the **Add Registry** link on the **Image Registries** page.

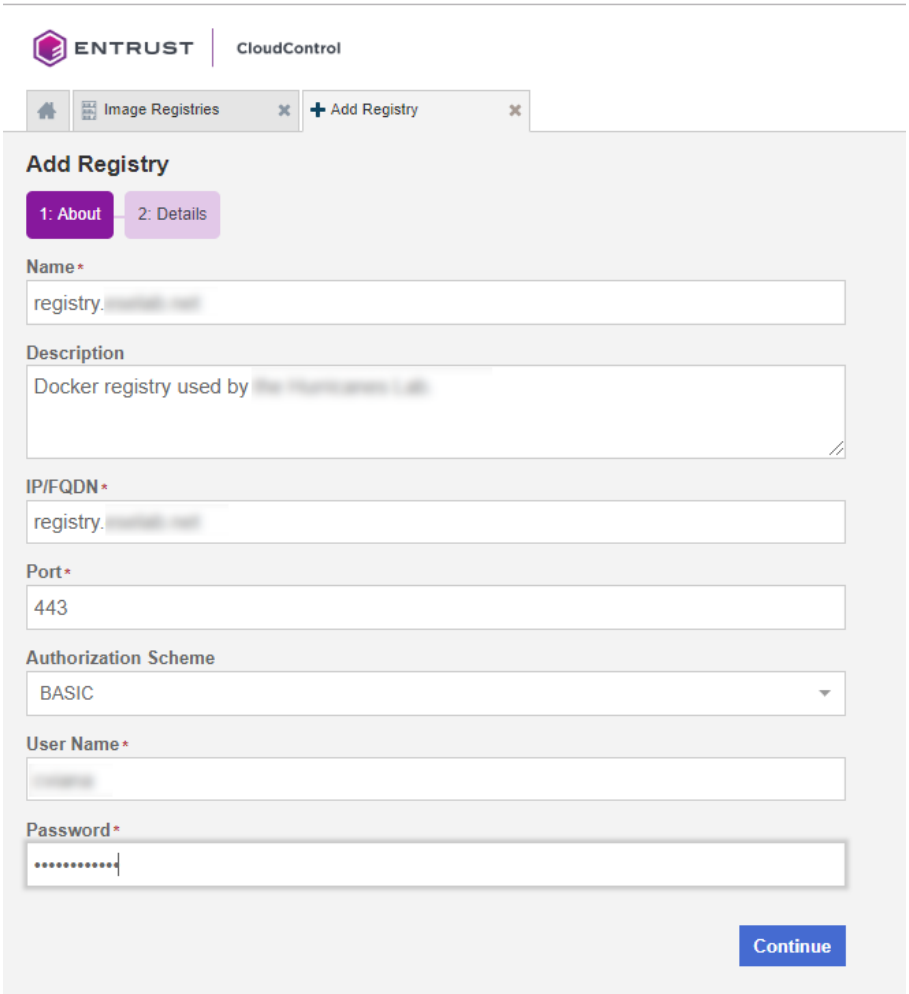
3. On the **Add Registry** page, in the **About** tab enter the following information:
   a. **Name** - Name of the registry
   b. **IP/FQDN** - Enter the IP Address or FQDN for the registry.
   c. **Port** - Enter the registry port used in configuration of the local registry.
   d. **Authorization Schema** - Choose one of the following to use for authorization: **BASIC** or **OAUTH**.

e. **User** - Enter the username for the registry.

f. **Password** - Enter the password for this user.

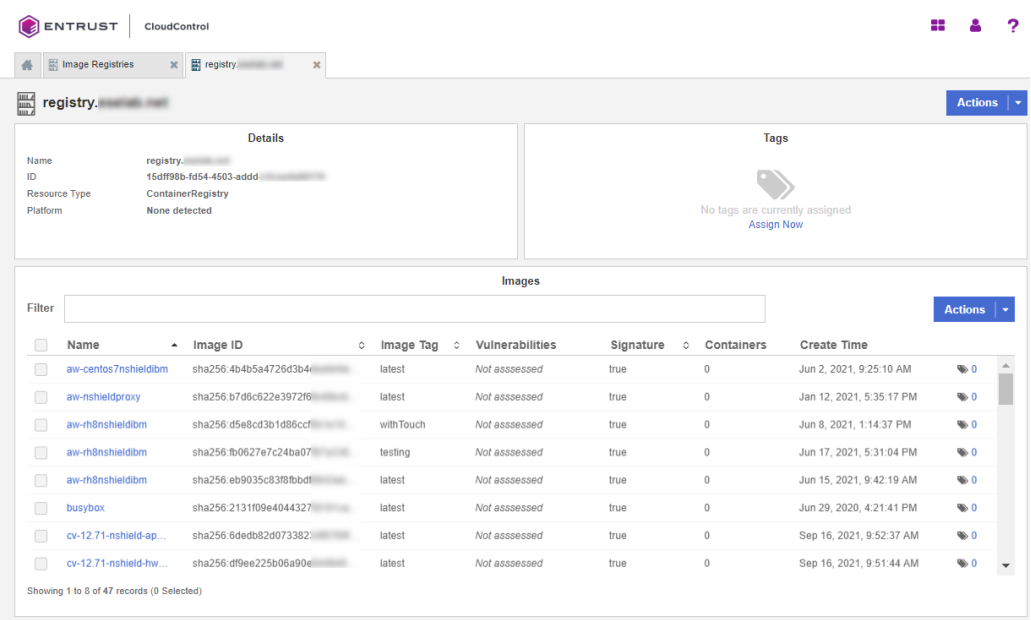
g. **Description** - Enter an optional description.

4. Select **Continue**.




If certificate authority has not already been added, the system prompts the user to add one.

1. In the **Missing Certificate Authority** window, select **Install Certificate Authority Now**.
2. On the **Install Certificates** page, do one of the following:
   a. Select **Import** and then select **Browse** to locate the certificate file.
   b. Select **Enter Text**, then paste the contents of the certificate as plain text into **Certificate Data**.
3. Select **Continue**.
4. On the **Details** page, monitor the process.
5. Select **Continue** to view the dashboard for the newly added registry.




Now create an Image Deployment Control Policy, see Image Deployment Control Policy.

## 2.13. Image Deployment Control Policy

The Image Deployment Control Policy controls how images are deployed in the Tanzu Kubernetes cluster managed by CloudControl. As part of a Trust Manifest, it allows the user to determine what images from a registry are safe to be added to protected clusters.

CloudControl enforces image security using Image Deployment Control Policies, which are comprised of one or more deployment rules:

### 2.13.1. Private registry rules

- Private Registries

  Allows the user to enter a list of private registries, either onboarded or not, to be evaluated with the trust manifest. Only the images from registries listed in the **Allowed Registries** section are evaluated to see if they can be deployed. Images from all other registries will be denied.

- Signature Rule

  Allows the user to deny images that do not have an associated digital signature.

- Attributes Rule

  Allows the user to deny or deploy images based on their image ID or image name.

- Vulnerabilities Rule

  Allows the user to deny or deploy images based on CVSS scores or specific CVEs.

### 2.13.2. Public registry rules

A public registry rule enables images from public registries to be deployed in the environment without any evaluation.

Entrust strongly recommends leaving the public registry rule set to ENABLED and not allowing public registries to be deployed. If a public registry must be used, then leave the rule set to ENABLED and enter that specific registry into the allowed registries. This will allow only that specific registry image to be deployed, and will prevent all other public registry images from deployment.

### 2.13.3. Other considerations

Rules can either have a True or False value and can also include a 'stop processing' clause. Deployment rules in a policy are evaluated in the following order:

- If False, the image will not be deployed, and no further rules are evaluated.
- If True, AND there is a 'stop processing' clause, the image is allowed to be deployed and no further rules are evaluated.
- If True, the next rule in the policy is evaluated. If this is the only rule, then the image is allowed to be deployed.
- If all rules are True, then the image is allowed to be deployed.

Entrust recommends always using the image SHA as it is a unique identifier for the images. Pods can be created by using either an image name with tag, or an image name with SHA, and in many cases images with the same SHA could have been tagged with different tags. For example, a single image named **TestImage** could have different tags, **TestImage:3**, **TestImage:4**, and **TestImage:5**, but all these images will have the same SHA because the underlying image is the same for all three of them.

When any Image Deployment Control Policy rule is created, use the image name with SHA to ensure that the intended image is evaluated no matter what tags are there. When an image name with tag is used, such as **TestImage:3**, then only the image that matches that specific tag will be selected. The other images, TestImage:4 and TestImage:5 will not be

evaluated.

### 2.13.4. Create an Image Deployment Control Policy

This section will show how an Image Deployment Control Policy is used to control which images are allowed/denied in the deployment.

For more information refer to Creating a Deployment Control Trust Manifest from the CloudControl GUI in the online documentation.

1. From the **Home** tab, select **Security > Trust Manifests**.
2. On the **Manage Trust Manifests** page, select **Actions > Create Trust Manifest**.
3. On the **Details** tab of the **Create Trust Manifest** page, enter the **Name** and optional **Description** for the trust manifest.
4. For **Policy Type**, select **Deployment Control**.
5. In the **Deployment Control Rules: Private registries** section:
    a. For **Allowed Registries**, enter the registries to allow. That is:
        - Enter the registry that was onboarded: **xxxxx.yyyy.zzz**.
        - Registries can be existing onboarded registries or registries that will be onboarded later.
        - Registries that have not been onboarded are depicted with a yellow warning icon.
6. In the **Deployment Control Rules: Rules** section:
    a. For **Signature Rule**, select either **Enabled** or **Disabled**. This determines whether to deny an image when no signature is present.
7. In the **Deployment Control Rules: Rules** section:
    a. For **Attribute Rule**, select **Enabled** or **Disabled** to determine whether to evaluate using attributes, and then complete the following:
        i. **Name** - Enter the name of the rule. The name cannot contain any special characters.
        ii. **Exemption List** - Deploy on Match

        Select **ENABLED** or **DISABLED** to determine whether to use this when evaluating.

        If **ENABLED**, use the + and - symbols to add the following criteria:

        - **Image ID** - Enter the image ID in SHA format to match.
        - **Image Name** - Enter the Name and Tag Regex to match.

If there is a match, the image will immediately be deployed, and no other deployment policy rules will be evaluated.

If there is no match, continue to the next enabled step.

If there are no other steps, continue to the next rule in the deployment policy.

iii. **Whitelist** - Deny on No Match

Select **ENABLED** or **DISABLED** to determine whether to use this when evaluating.

If **ENABLED**, use the + and - symbols to add the following criteria:

- **Image ID** - Enter the image ID in SHA format to match.
- **Image Name** - Enter the Name and Tag Regex to match.

If there is no match, the image will immediately be denied, and no other deployment policy rules will be evaluated.

If there is a match, continue to the next enabled step.

If there are no other steps, continue to the next rule in the deployment policy.

iv. **Blacklist** - Deny on Match

Select **ENABLED** or **DISABLED** to determine whether to use this when evaluating.

If **ENABLED**, use the + and - symbols to add the following criteria:

- **Image ID** - Enter the image ID in SHA format to match.
- **Image Name** - Enter the Name and Tag Regex to match.

If there is a match, the image will immediately be denied, and no other deployment policy rules will be evaluated.

If there is no match, continue to the next rule in the deployment policy.

b. Optional. In the **Public Registries** section, add public registries to be deployed without any evaluation.

Entrust recommends leaving this section enabled, but not entering any values for **Allowed Registries**.

8. Select one of the following:
   - **Validate** - Validate the draft or existing trust manifest.

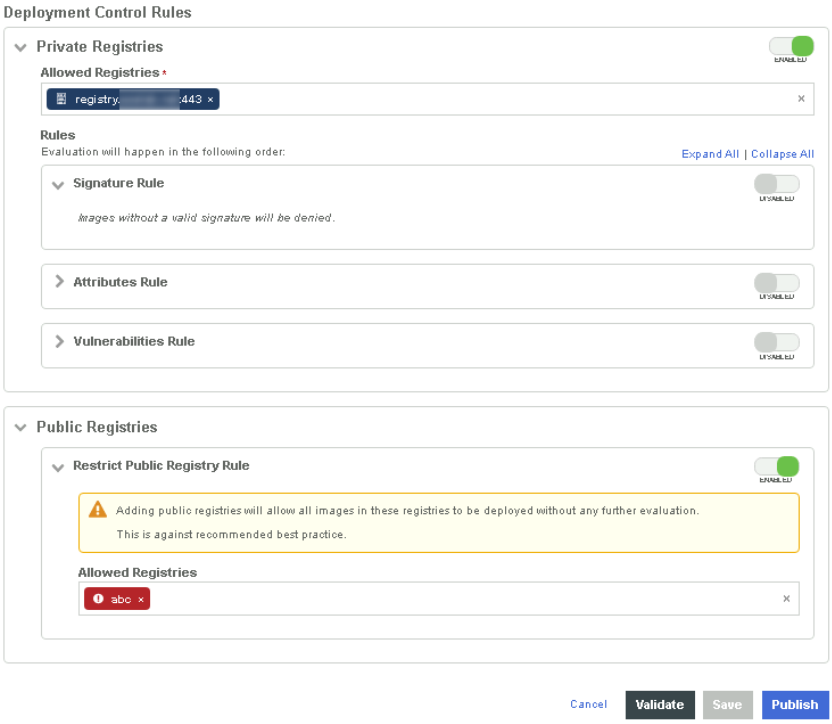
- **Save** - Save the trust manifest as a draft.
- **Publish** - Publish the trust manifest.

## 2.13.5. Image Deployment Control Policy examples

In this example, an Image Deployment Control Policy is created. It will only allow images that are in the private registry that was onboarded earlier. Any other image that is not in the private registry will be blocked and will not run.

To be able to publish the policy:

- The **Restrict Public Registry** rule will be enabled.
- A fake registry name **abc** will be added to the exception list. This will force the policy to only allow images in the private registry.




After this policy is published, it is possible to deploy an image that is not in the registry.

For example, using the `pod.yaml` file again:

```
apiVersion: v1
kind: Pod
metadata:
    name: tutum-centos3
spec:
    containers:
    - name: tutum-ssh-server3
      image: tutum/centos
```

The YAML file is not using an image from the private registry.

```
% kubectl create -f pod.yaml

Error from server (Forbidden): error when creating "pod.yaml": admission webhook "deploymentcontrol.hytrust.com"
denied the request: Permission Denied by HyTrust Security Policy.
```

In this example, the pod was not deployed because it uses an image from the **tutum** registry.

To use an image from a private registry using a different YAML file. For example, `pod-internal-registry.yaml` with the following content:

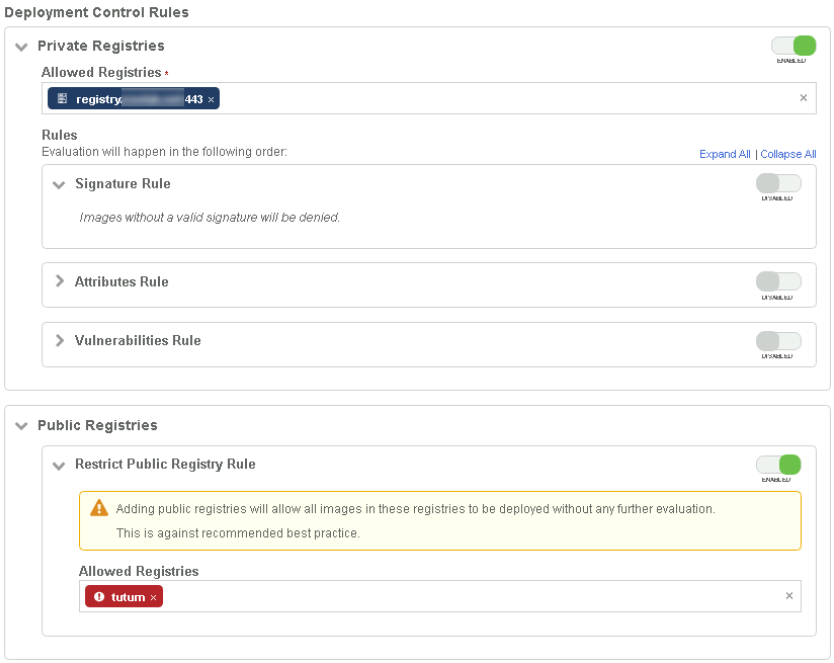
```
apiVersion: v1
kind: Pod
metadata:
    name: internal-registry-centos
spec:
    imagePullSecrets:
    - name: regcred
    containers:
    - name: internal-registry-centos
      image: xxxxxx.yyyy.zzz/cv-centos:latest
```

Assuming the ImagePullSecret **regcred** which has the credentials to access the private registry has been created, create the pod:

```
% kubectl create -f pod-internal-registry.yaml

pod/internal-registry-centos created
```

The deployment is successful, because the Image Deployment Control Policy allows images from the private registry.

Add the **tutum** external registry to the external registry exception list, so an external image can also be deployed. For example:




Create the pod using `pod.yaml` file again:

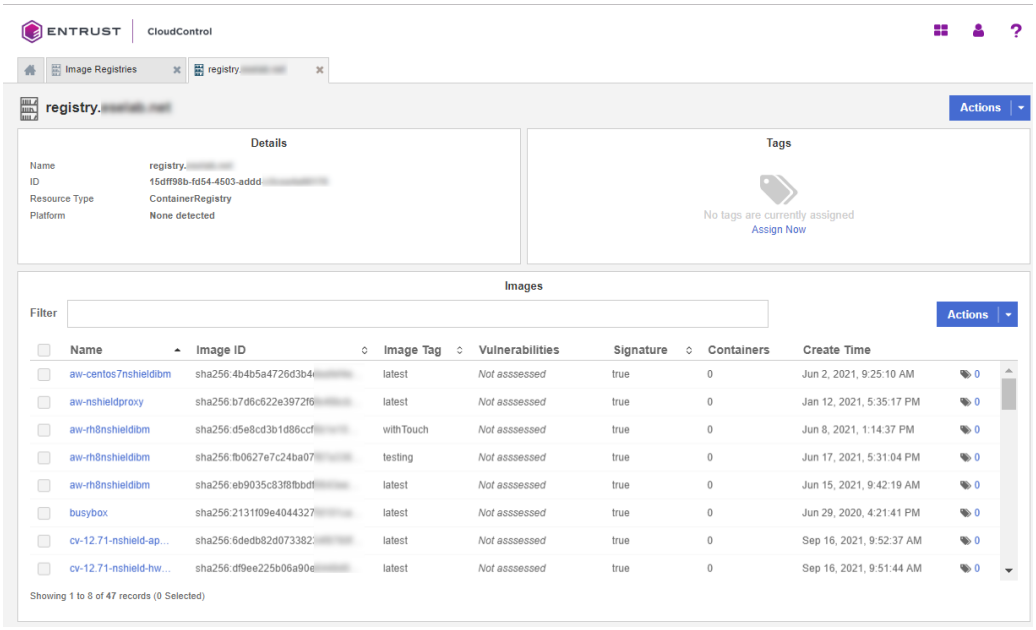
```
% kubectl create -f pod.yaml

pod/tutum-centos3 created
```

The deployment policy is working as designed.

Expand the policy by further restricting what images can be deployed from the private registry.

To determine what images are in the private registry:

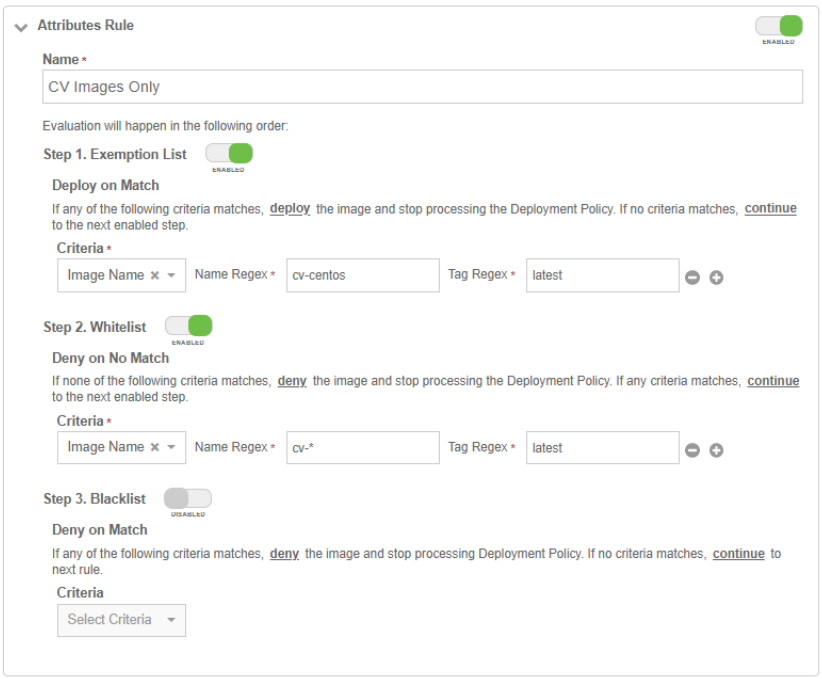
1. In the **Home** tab, Select **Inventory > Image Registries** and select the private registry that was onboarded.




2. Look at the images that are available in the registry.

Change the deployment policy to only allow images with name **cv-centos** and with the **latest** tag. Any other image in the private registry that does not match this requirement in the name will be blocked.

To do this, add an **Attribute Rule** to the deployment policy. For example:




Now try to deploy the **busybox** image in the private registry. To do this, create a file called `pod-internal-busybox.yaml` with the following content:

```
apiVersion: v1
kind: Pod
metadata:
    name: internalregistry-busybox
spec:
    imagePullSecrets:
    - name: regcred
    containers:
    - name: internalregistry-busybox-test
      image: xxxxxx.yyyy.zzz/busybox
```

When this is deployed, it should fail and deny the deployment. For example:

```
% kubectl create -f pod-internal-busybox.yaml

Error from server (Forbidden): error when creating "pod-internal-busybox.yaml": admission webhook
"deploymentcontrol.hytrust.com" denied the request: Permission Denied by HyTrust Security Policy.
```

Now deploy the **cv-centos** image in the private registry. To do this, use the `pod-internal-registry.yaml` file again.

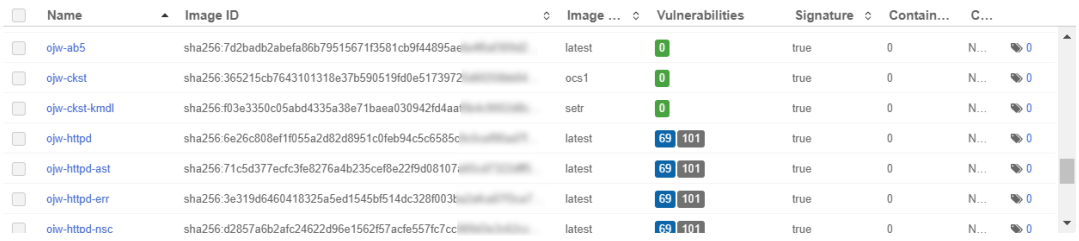
```
% kubectl create -f pod-internal-registry.yaml

pod/internal-registry-centos created
```

The deployment is successful.

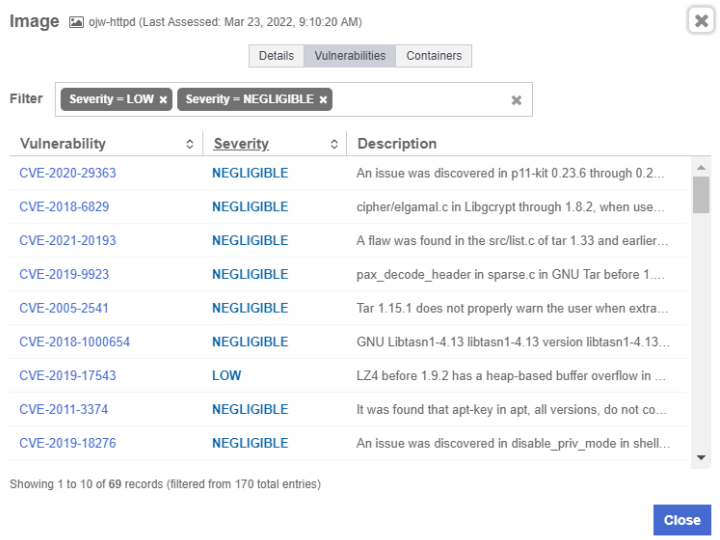
This demonstrates how to harden the Image Deployment Control Policy.

## 2.13.6. Image Deployment Control Policy based on vulnerabilities

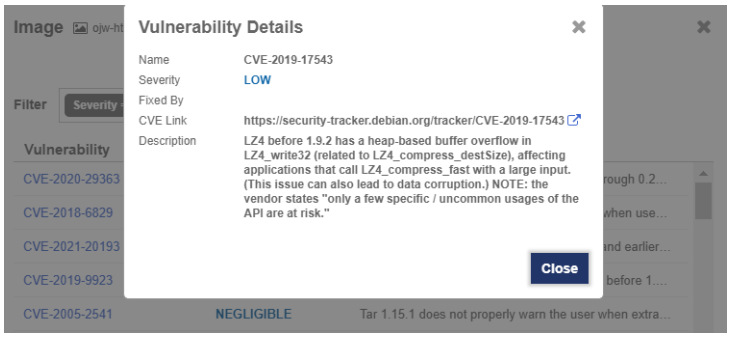
When onboarding a private registry into CloudControl, the system checks the number of vulnerabilities in each image in the registry. Below is an example of a private registry when it was onboarded.




For each image, CloudControl collects the number of vulnerabilities and their types. Select a link in the **Vulnerabilities** column to view a pop-up with tabs that include the details.




The **Vulnerabilities** tab lists each vulnerability with their CVE, severity, and description. Select the CVE link to view details about the CVE. For example:




With this information on hand, create/modify the Image Deployment Control Policy to allow/deny deployments based on the number of vulnerabilities an image contain.

For example, modify the current Image Deployment Control Policy to allow any image from the private registry, but only if it has no vulnerabilities. This requires the use of two images from the private registry:

- **cv-centos** - which has no vulnerability.
- **ojw-httpd** - which has some vulnerabilities.

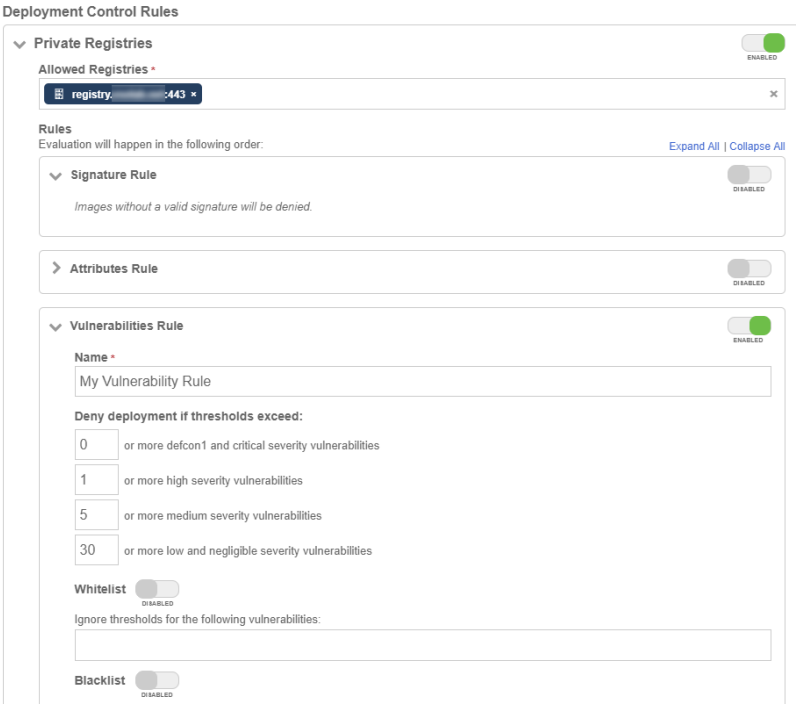
### 2.13.6.1. Modify the Image Deployment Control Policy

Modify the policy by disabling the **Attributes Rule** and enabling the **Vulnerabilities Rule**. The default thresholds will be appropriate for testing.

1. From the **Home** tab, select **Security > Trust Manifests**.
2. On the **Manage Trust Manifests** page, select the deployment control policy created earlier.

3. Select **Edit**.
4. Disable the **Attribute Rule** section under **Rules**, under **Private Registries**.
5. Enable the **Vulnerability Rule** section.
    a. Give a name for the rule, for example **My Vulnerability Rule**.
    b. Take the defaults for the **Deny deployment thresholds** values.

    The **ojw-httpd** image has more than 30 low severity vulnerabilities.
6. Select **Validate** and the select **Apply**.




Test the policy, see Test the policy.

### 2.13.6.2. Test the policy

To test the policy, attempt to deploy the **cv-centos** image. Use the same `pod-internal-registry.yaml` file again.

```
% kubectl create -f pod-internal-registry.yaml

pod/internal-registry-centos created
```

The deployment succeeds.

Now attempt to deploy the **ojw-httpd** image. To do this, create a file called `pod-internal-cve.yaml` with the following content:

```
apiVersion: v1
kind: Pod
metadata:
    name: internalregistry-ojw-httpd
spec:
    imagePullSecrets:
    - name: regcred
    containers:
    - name: internalregistry-ojw-httpd
      image: xxxxxx.yyyy.zzz/ojw-httpd:latest
```

It should fail and be denied.

```
% kubectl create -f pod-internal-cve.yaml

Error from server (Forbidden): error when creating "pod-internal-cve.yaml": admission webhook
"deploymentcontrol.hytrust.com" denied the request: Permission Denied by HyTrust Security Policy.
```

Examine the logs to see the denial.

1. Go to the **Home** tab, select **Security > Log Analysis**.
2. Select the record that shows the **Deny** status to see the reason for the denial.
3. Look for something like this:

This feature from CloudControl allows the user to put in place image deployment control policies that can harden the organization deployment requirements and tailor this capability according to the organization needs.

## 2.14. Configuration hardening

Configuration hardening allows the user to improve the security posture of Kubernetes environments by hardening the configuration to meet the company's specific security policy and industry best practices. By automating the hardening process, the user can reduce the operational burden during a compliance audit.

With CloudControl, the user can:

- Create and customize templates to use in configuration hardening checks.
- Assess and remediate your environments against the configuration hardening checks

defined in the templates.

- Review dashboards, reports and alerts to monitor the results of assessments and remediations.

### 2.14.1. About templates

CloudControl uses templates to support all configuration hardening activities. CloudControl supports the following types of templates:

- **Catalog templates** - Read-only collection of hardening operations. There is a Kubernetes operations catalog of templates that can be used.
- **System templates** - Read-only collection of operations derived from a catalog template for a given compliance standard
- **Custom templates** - Templates created by users. In most cases, they are copied or cloned from existing system or catalog templates. Custom templates can be modified and used in configuration hardening policies.

CloudControl also includes sample custom templates that can immediately be used in a policy.

Templates can contain both assessment and remediation hardening operations. Review all operations in the template to ensure that any parameter values are set appropriately for the infrastructure requirements.

### 2.14.2. About policies

Configuration hardening policies are used to run custom templates. Each policy associates a template with one or more resources or tag-based resource configurations and can be run manually or as a scheduled activity. Policies can either assess or remediate a resource but cannot do both.

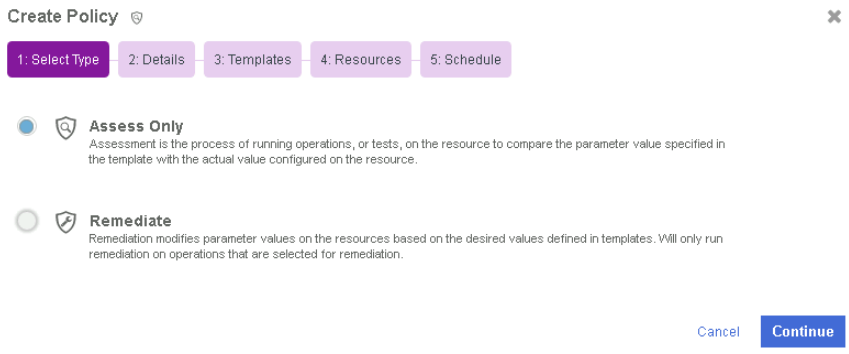
### 2.14.3. More information

Consult the online documentation for more information on Configuration Hardening.

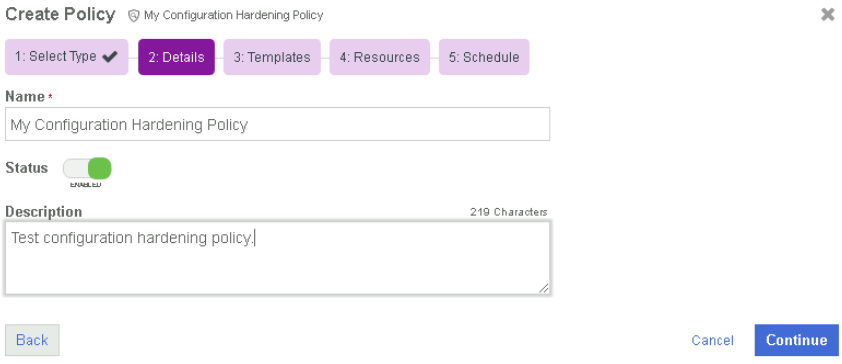
### 2.14.4. Creating a configuration hardening policy example

For this guide a configuration hardening policy will be created based on one of the templates available.

1. From the **Home** tab, select **Security > Configuration Hardening**.
2. On the **Configuration Hardening Management** page, select the **Policies** tab.
3. Select the **Create (+)** button.
4. In the **Create Policy** wizard on the **Select Type** page, select the type of policy that you want to create. This can be one of the following:
    - **Assess Only** - Runs operations on the host to compare the parameter values specified in the template with the actual values on the host.
    - **Remediate Only** - Modifies the parameter values on the host in order to match the values specified in the templates.
5. Select **Assess Only**.



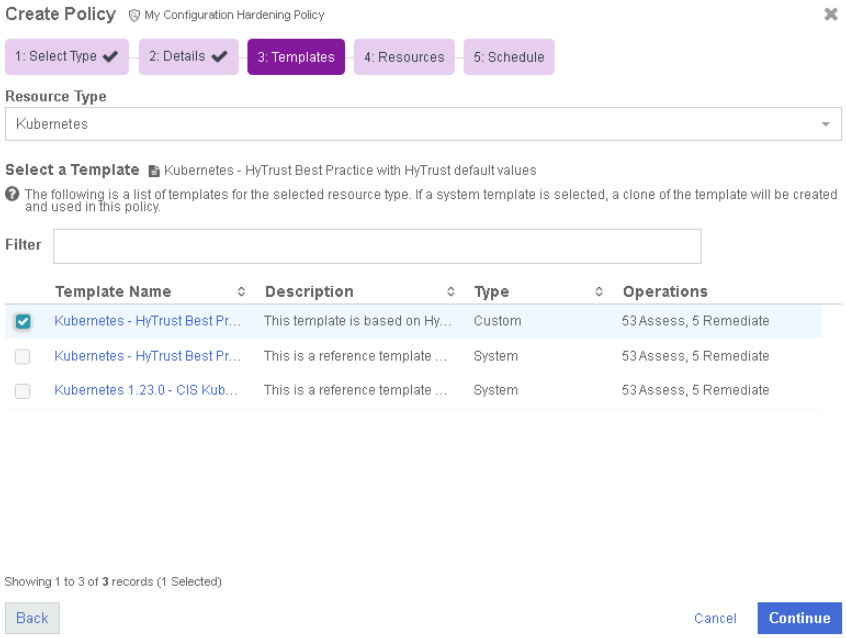

6. Select **Continue**.
7. On the **Details** page, enter the name and optional description of the policy, and specify whether the policy is enabled.




8. Select **Continue**.
9. On the **Templates** page, select the **resource type** for the policy. This can be one of the following:
    - **AWS Account** - Runs AWS-related templates against your AWS environment.
    - **ESXi** - Runs vSphere related templates against your ESXi hosts.
    - **Kubernetes** - Runs Kubernetes related templates against your Kubernetes environment.

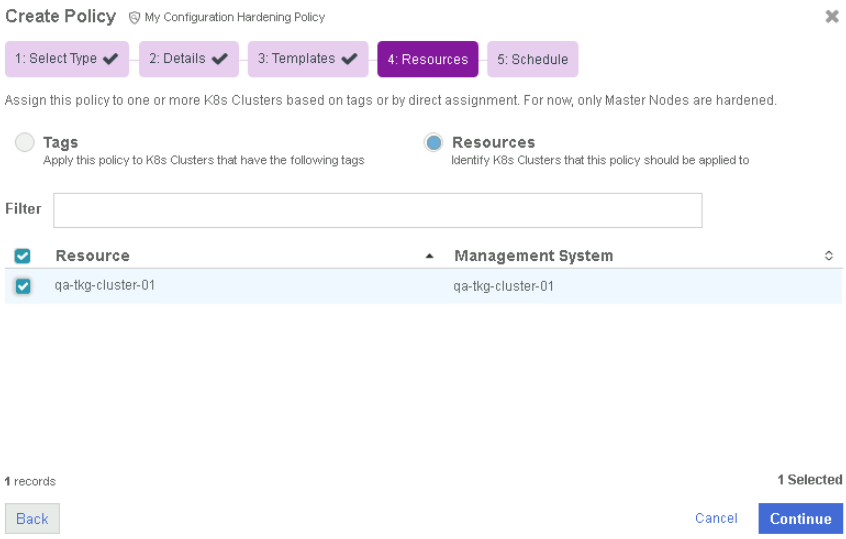
- **NSXDataCenter** - Runs NSX-T related templates against your NSX-T environment.

10. Select **Kubernetes** as the template to run against your environment.
11. The template list displays the name, description and type of operations Select a template that contains the type of operations that you selected for the policy.
12. Select **Kubernetes - HyTrust Best Practice with HyTrust default values** template. This template is used in the example.



See the online documentation for more information on Creating a Custom Template.

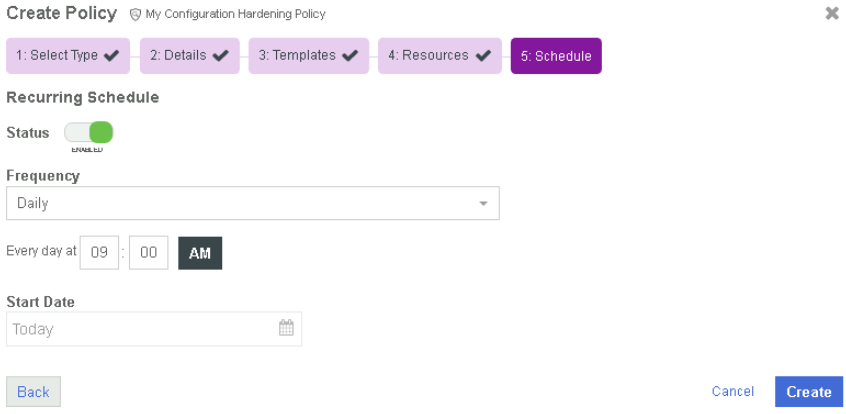
13. Select **Continue**.
14. On the **Assignments** page, select one of the following and choose the resources to apply the policy to:
    - **Tags** - Select the **Tags** radio button and then choose a tag or tags assigned to the resource. Select the **+** icon to assign more tags to the resource. If there are no tags assigned, select the **Assign Tags Now** link.
    - **Resources** - Select **Specific Resources** and then choose one or more resources.
15. Select **Resources** for the resources to apply the policy to.
16. Select the Kubernetes cluster that has been onboarded.



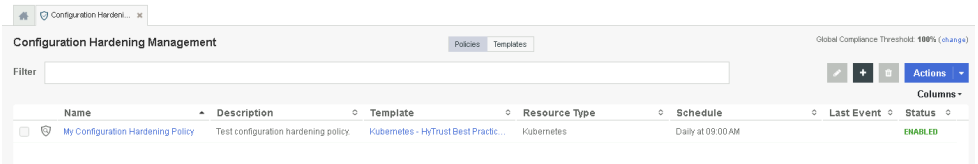

17. Select **Continue**.
18. On the **Schedule** page, enable a recurring schedule if required.

    If enabled, select the type of schedule to use to run the policy and specify the start date. This can be one of the following:

    - **Daily** - The policy will run every day at the specified time.
    - **Hourly** - The policy will run periodically throughout the day, based on the defined schedule.
    - **Weekly** - The policy will run on every day that you select at the specified time.




19. Select **Create**.
20. The new policy appears on the **Policies** tab.



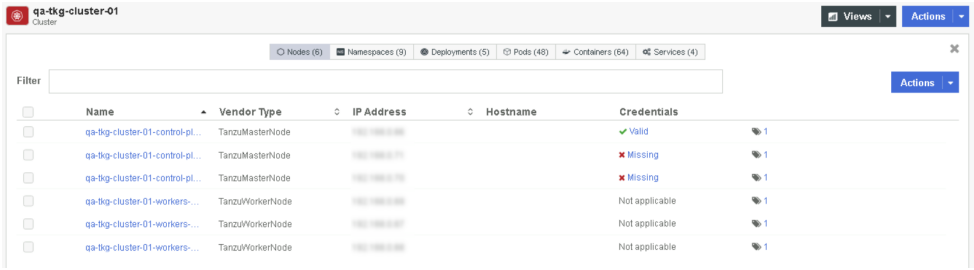
## 2.14.5. Adding credentials to Tanzu master nodes

This is required to be able to run the configuration hardening policy. Follow the instructions, as described here, on how to SSH to Tanzu Kubernetes Cluster Nodes as the System User Using a Password.

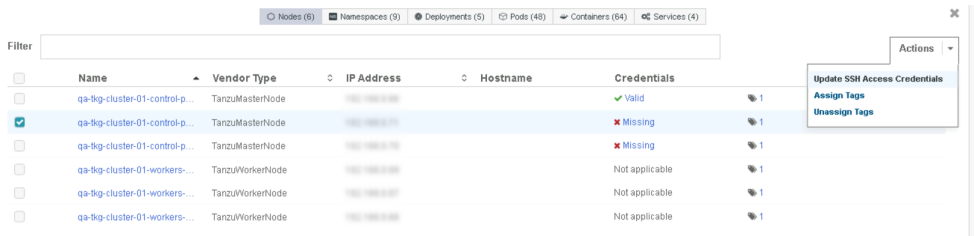
After obtaining the password, save it. This password is used to store the credentials of the master nodes in CloudControl. For example:

```
Credentials : vmware-system-user / zeaIHcnKYYINzqF6TQ6D3c4gSuVUtJizl35uxKZ+Ks8=
```

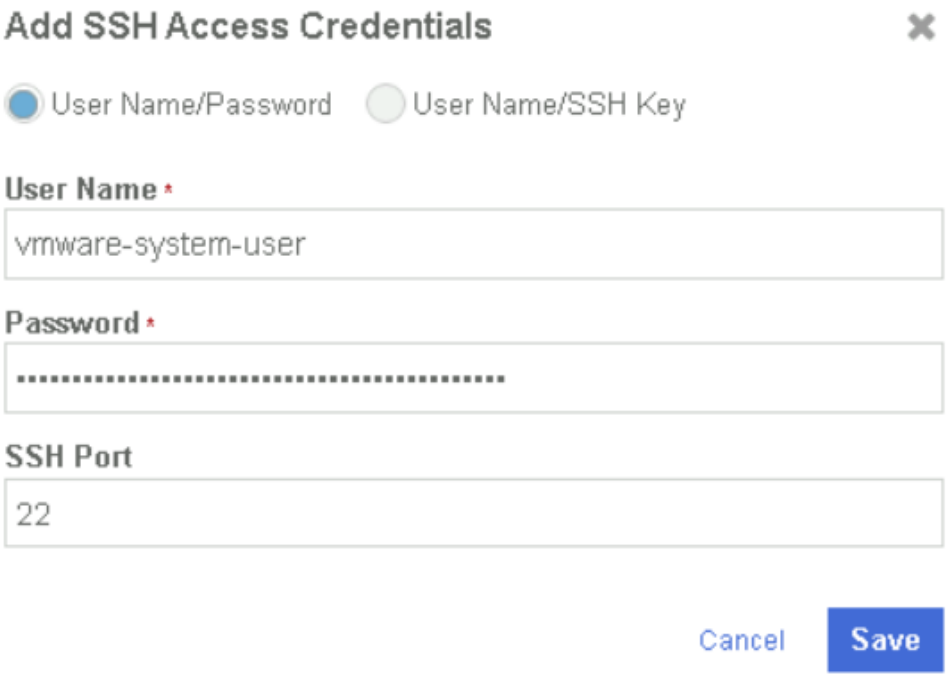
1. Navigate to the Tanzu cluster.
2. From the **Home** tab, select **Inventory > Kubernetes Cluster**.
3. Select the link for the Tanzu cluster. In this case the **qa-tkg-cluster-01**.
4. On the **qa-tkg-cluster-01** page, select the **Nodes** tab. Add any missing credentials to the master nodes.




5. Select the master node and then from the **Actions** Menu select **Update SSH Access Credentials**.




6. In the **Add SSH Credentials** window, enter the **User Name** and **Password** saved earlier.

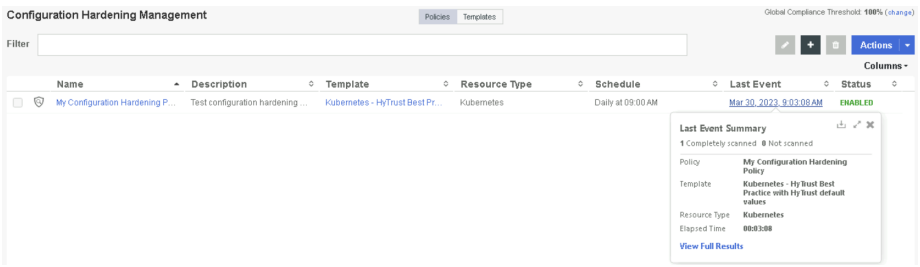



7. Select **Save**.
8. Do this for all the master nodes in the cluster.

## 2.14.6. Run the new configuration hardening policy

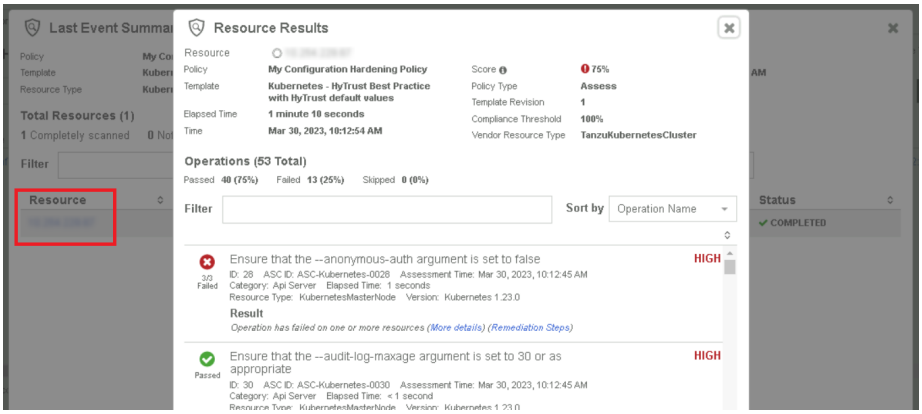
When running a remediation policy, the assessment policy automatically runs immediately following its completion. This ensures that the compliance score is updated with the new percentage.

1. From the **Home** tab, select **Security > Configuration Hardening**.
2. On the **Configuration Hardening Management** page, select the **Policies** tab.
3. Select the policy that you want to run.
4. Select **Actions > Run Now**.
5. You can view the results by selecting the link in the **Last Event** column after the configuration hardening policy finishes running.
6. Select **View Full Results**.



When viewing the full results, if you see a warning message indicating that it "Failed to Assess one or more resource(s)", make sure you have added the credentials to all Tanzu master nodes to perform compliance. See Adding credentials to Tanzu master nodes for more details.

7. In the **Last Event Summary** Tab, select **Resource**.




8. Check the results to see whether each test in the policy **Passed** or **Failed**. Select the **More details** link to get details on the result for each of the master nodes in the cluster. For **Failed** tests, select the **Remediation Steps** link to get information on how to remediate the problem.

# Chapter 3. Additional resources and related products

## 3.1. CloudControl

## 3.2. Entrust products

## 3.3. nShield product documentation