Red Hat Advanced Cluster Security for Kubernetes 3.70

Installing

Installing Red Hat Advanced Cluster Security for Kubernetes

Last Updated: 2022-12-06

Red Hat Advanced Cluster Security for Kubernetes 3.70 Installing

Installing Red Hat Advanced Cluster Security for Kubernetes

Legal Notice

Copyright © 2022 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at
http://creativecommons.org/licenses/by-sa/3.0/
. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This document describes how to install Red Hat Advanced Cluster Security for Kubernetes by using the Operator, Helm charts, or the roxctl CLI.

Table of Contents

CHAPTER 1. PREREQUISITES FOR INSTALLING RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES 4
1.1. GENERAL REQUIREMENTS 4
1.2. PREREQUISITES FOR INSTALLING CENTRAL 5
Memory and storage requirements 5
Sizing guidelines 6
1.3. PREREQUISITES FOR INSTALLING SCANNER 6
Memory and storage requirements 6
1.4. PREREQUISITES FOR INSTALLING SENSOR 6
Memory and storage requirements 6
1.5. PREREQUISITES FOR INSTALLING ADMISSION CONTROLLER 6
Memory and storage requirements 7
1.6. PREREQUISITES FOR INSTALLING COLLECTOR 7
Memory and storage requirements 7

CHAPTER 2. INSTALLATION PLATFORMS AND METHODS 8
2.1. INSTALLATION METHODS FOR DIFFERENT PLATFORMS 8

CHAPTER 3. INSTALLING BY USING AN OPERATOR 10
3.1. INSTALLING THE RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES OPERATOR 11
3.2. INSTALLING CENTRAL 12
3.3. VERIFYING CENTRAL INSTALLATION 13
3.4. CENTRAL CONFIGURATION OPTIONS 14
3.4.1. Central settings 14
3.4.2. Scanner settings 16
3.4.3. General and miscellaneous settings 16
3.5. GENERATING AN INIT BUNDLE 17
3.5.1. Generating an init bundle by using the RHACS portal 17
3.5.2. Generating an init bundle by using the roxctl CLI 18
3.5.3. Additional resources 18
3.6. CREATING RESOURCES BY USING THE INIT BUNDLE 18
3.7. INSTALLING SECURED CLUSTER SERVICES 19
3.8. SECURED CLUSTER CONFIGURATION OPTIONS 20
3.8.1. Required Configuration Settings 20
3.8.2. Admission controller settings 21
3.8.3. Scanner configuration 22
3.8.4. Image configuration 23
3.8.5. Per node settings 23
3.8.6. Taint Tolerations settings 24
3.8.7. Sensor configuration 24
3.8.8. General and miscellaneous settings 24
3.9. VERIFYING INSTALLATION 25
3.10. ADDING A NEW CLUSTER TO RHACS 26

CHAPTER 4. INSTALLING USING HELM CHARTS 27
4.1. INSTALLING QUICKLY USING HELM CHARTS 27
4.1.1. Adding the Helm chart repository 27
4.1.2. Installing the central-services Helm chart without customization 28
4.1.3. Generating an init bundle 29
4.1.3.1. Generating an init bundle by using the roxctl CLI 29
4.1.4. Installing the secured-cluster-services Helm chart without customization 29
4.1.5. Verifying installation 30

4.1.6. Additional resources 31
4.2. INSTALLING WITH CUSTOMIZATIONS USING HELM CHARTS 31
4.2.1. Adding the Helm chart repository 32
4.2.2. Configuring the central-services Helm chart 32
4.2.2.1. Private configuration file 33
4.2.2.1.1. Image pull secrets 33
4.2.2.1.2. Proxy configuration 34
4.2.2.1.3. Central 34
4.2.2.1.4. Scanner 35
4.2.2.2. Public configuration file 36
4.2.2.2.1. Image pull secrets 36
4.2.2.2.2. Image 37
4.2.2.2.3. Environment variables 37
4.2.2.2.4. Additional trusted certificate authorities 37
4.2.2.2.5. Central 38
4.2.2.2.6. Scanner 40
4.2.2.2.7. Customization 41
4.2.2.2.8. Advanced customization 43
4.2.3. Installing the central-services Helm chart 43
4.2.3.1. Changing configuration options after deploying the central-services Helm chart 44
4.2.4. Generating an init bundle 44
4.2.4.1. Generating an init bundle by using the roxctl CLI 44
4.2.4.2. Generating an init bundle by using the RHACS portal 45
4.2.5. Configuring the secured-cluster-services Helm chart 46
4.2.5.1. Configuration parameters 46
4.2.5.1.1. Environment variables 53
4.2.6. Installing the secured-cluster-services Helm chart 53
4.2.6.1. Changing configuration options after deploying the secured-cluster-services Helm chart 54
4.2.7. Verifying installation 54

CHAPTER 5. INSTALLING BY USING THE ROXCTL CLI 56
5.1. INSTALLING THE ROXCTL CLI 56
5.2. INSTALLING THE ROXCTL CLI ON LINUX 56
5.2.1. Installing the roxctl CLI on macOS 57
5.2.2. Installing the roxctl CLI on Windows 57
5.3. INSTALLING CENTRAL 58
5.3.1. Using the interactive installer 58
5.3.2. Running the Central installation scripts 59
5.4. INSTALLING SCANNER 60
5.5. INSTALLING SENSOR 61
5.6. VERIFYING INSTALLATION 62
5.7. ADDITIONAL RESOURCES 63

CHAPTER 6. UNINSTALLING RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES 64
6.1. DELETING NAMESPACE 64
6.2. DELETING GLOBAL RESOURCES 64
6.3. DELETING LABELS AND ANNOTATIONS 65

CHAPTER 1. PREREQUISITES FOR INSTALLING RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES

1.1. GENERAL REQUIREMENTS

To install Red Hat Advanced Cluster Security for Kubernetes, you must have:

- OpenShift Container Platform version 4.5 or later for an OpenShift Container Platform installation.

WARNING

You must not install Red Hat Advanced Cluster Security for Kubernetes on:

  - Amazon Elastic File System (Amazon EFS). Use the Amazon Elastic Block Store (Amazon EBS) with the default **gp2** volume type instead.
  - Older CPUs that do not have the Streaming SIMD Extensions (SSE) 4.2 instruction set. For example, Intel processors older than *Sandy Bridge* and AMD processors older than *Bulldozer*. (These processors were released in 2011.)

- Cluster nodes with a supported operating system. For more information, see the Red Hat Advanced Cluster Security for Kubernetes Support Policy.
  - **Operating system**: Amazon Linux, CentOS, Container-Optimized OS from Google, Red Hat Enterprise Linux CoreOS (RHCOS), Debian, Red Hat Enterprise Linux (RHEL), or Ubuntu.
  - **Processor and memory**: 2 CPU cores and at least 3GiB of RAM.

NOTE

For deploying Central, use a machine type with 4 or more cores and apply scheduling policies to launch Central on such nodes.

- Persistent storage by using persistent volume claim (PVC).

IMPORTANT

You must not use Ceph FS storage with Red Hat Advanced Cluster Security for Kubernetes. Red Hat recommends using RBD block mode PVCs for Red Hat Advanced Cluster Security for Kubernetes.

  - Use Solid-State Drives (SSDs) for best performance. However, you can use another storage type if you do not have SSDs available.

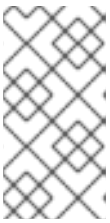
- Helm command-line interface (CLI) v3.2 or newer, if you are installing or configuring Red Hat Advanced Cluster Security for Kubernetes using Helm charts. Use the **helm version** command to verify the version of Helm you have installed.
- The OpenShift Container Platform CLI (**oc**).
- You must have the required permissions to configure deployments in the Central cluster.
- You must have access to the Red Hat Container Registry. For information about downloading images from **registry.redhat.io**, see Red Hat Container Registry Authentication .

1.2. PREREQUISITES FOR INSTALLING CENTRAL

A single containerized service called Central handles data persistence, API interactions, and user interface (Portal) access.

Central requires persistent storage:

- You can provide storage with a persistent volume claim (PVC).



NOTE

You can use a hostPath volume for storage only if all your hosts (or a group of hosts) mount a shared file system, such as an NFS share or a storage appliance. Otherwise, your data is only saved on a single node. Red Hat does not recommend using a hostPath volume.

- Use Solid-State Drives (SSD) for best performance. However, you can use another storage type if you do not have SSDs available.
- If you use a web proxy or firewall, you must configure bypass rules to allow traffic for the **definitions.stackrox.io** and **collector-modules.stackrox.io** domains and enable Red Hat Advanced Cluster Security for Kubernetes to trust your web proxy or firewall. Otherwise, updates for vulnerability definitions and kernel support packages will fail.
  Red Hat Advanced Cluster Security for Kubernetes requires access to:
  - **definitions.stackrox.io** for downloading updated vulnerability definitions. Vulnerability definition updates allow Red Hat Advanced Cluster Security for Kubernetes to maintain up-to-date vulnerability data when new vulnerabilities are discovered or additional data sources are added.
  - **collector-modules.stackrox.io** to download updated kernel support packages. Updated Kernel support packages ensure that Red Hat Advanced Cluster Security for Kubernetes can monitor the latest operating systems and collect data about the network traffic and processes running inside the containers. Without these updates, Red Hat Advanced Cluster Security for Kubernetes might fail to monitor containers if you add new nodes in your cluster or if you update your nodes' operating system.

NOTE

For security reasons, you should deploy Central in a cluster with limited administrative access.

Memory and storage requirements

The following table lists the minimum memory and storage values required to install and run Central.

Central	CPU	Memory	Storage
Request	1.5 cores	4 GiB	100 GiB
Limit	4 cores	8 GiB	100 GiB

Sizing guidelines

Use the following compute resources and storage values depending upon the number of nodes in your cluster.

Nodes	Deployments	CPU	Memory	Storage
Up to 100	Up to 1000	2 cores	4 GiB	100 GiB
Up to 500	Up to 2000	4 cores	8 GiB	100 GiB
More than 500	More than 2000	8 cores	12 - 16 GiB	100 - 200 GiB

1.3. PREREQUISITES FOR INSTALLING SCANNER

Red Hat Advanced Cluster Security for Kubernetes includes an image vulnerability scanner called Scanner. This service scans images that are not already scanned by scanners integrated into image registries.

Memory and storage requirements

Scanner	CPU	Memory
Request	1.2 cores	2700 MiB
Limit	5 cores	8000 MiB

1.4. PREREQUISITES FOR INSTALLING SENSOR

Sensor monitors your Kubernetes and OpenShift Container Platform clusters. These services currently deploy in a single deployment, which handles interactions with the Kubernetes API and coordinates with Collector.

Memory and storage requirements

Sensor	CPU	Memory
Request	1 core	1 GiB
Limit	2 cores	4 GiB

1.5. PREREQUISITES FOR INSTALLING ADMISSION CONTROLLER

The Admission controller prevents users from creating workloads that violate policies you configure.

Memory and storage requirements

By default, the admission control service runs 3 replicas. The following table lists the request and limits for each replica.

Admission controller	CPU	Memory
Request	.05 cores	100 MiB
Limit	.5 cores	500 MiB

1.6. PREREQUISITES FOR INSTALLING COLLECTOR

Collector monitors runtime activity on each node in your secured clusters. It connects to Sensor to report this information.

CAUTION

To install Collector on systems that have Unified Extensible Firmware Interface (UEFI) and that have Secure Boot enabled, you must use eBPF probes because kernel modules are unsigned, and the UEFI firmware cannot load unsigned packages. Collector identifies Secure Boot status at the start and switches to eBPF probes if required.

Memory and storage requirements

Collector	CPU	Memory
Request	.05 cores	320 MiB
Limit	.75 cores	1 GiB

NOTE

Collector uses a mutable image tag (**<version>-latest**), so you get support for newer Linux kernel versions more easily. There is no change in code, pre-existing kernel modules, or eBPF programs for image updates. Updates only add a single image layer with support for new kernel versions published after the initial release.

CHAPTER 2. INSTALLATION PLATFORMS AND METHODS

Red Hat Advanced Cluster Security for Kubernetes is supported on various platforms. This topic provides information for each platform and links to installation documentation.

2.1. INSTALLATION METHODS FOR DIFFERENT PLATFORMS

You can perform different types of installations on different platforms.

NOTE

Not all installation options are supported for all platforms, as shown in the following tables. Red Hat recommends that you do not use the **roxctl** install method unless you have a specific installation need that requires using this method.

Table 2.1. Self-managed platforms

Platform	Supported installation methods
Red Hat OpenShift Container Platform (OCP) 4.x	• Operator (recommended) • Helm • **roxctl**
Red Hat OpenShift Container Platform (OCP) 3.11.z	• Helm (recommended) • **roxctl**
Red Hat OpenShift Kubernetes Engine (OKE) 4.x	• Operator (recommended) • Helm • **roxctl**

Table 2.2. Managed services platforms

Platform	Supported installation methods
Red Hat OpenShift Dedicated (OSD)	• Operator (recommended) • Helm • **roxctl**

Platform	Supported installation methods
Azure Red Hat OpenShift (ARO)	• Operator (recommended) • Helm • **roxctl**
Red Hat OpenShift Service on AWS (ROSA)	• Operator (recommended) • Helm • **roxctl**
Amazon Elastic Kubernetes Service (Amazon EKS)	• Helm (recommended) • **roxctl**
Google Kubernetes Engine (Google GKE)	• Helm (recommended) • **roxctl**
Microsoft Azure Kubernetes Service (Microsoft AKS)	• Helm (recommended) • **roxctl**

CHAPTER 3. INSTALLING BY USING AN OPERATOR

Red Hat Advanced Cluster Security for Kubernetes (RHACS) installs a set of services on your OpenShift Container Platform or Kubernetes cluster. This section describes the installation procedure for installing Red Hat Advanced Cluster Security for Kubernetes on your OpenShift Container Platform or Kubernetes cluster by using an Operator.

Before you install:

- Understand Red Hat Advanced Cluster Security for Kubernetes architecture .
- Review the prerequisites for installing Red Hat Advanced Cluster Security for Kubernetes .

The Red Hat Advanced Cluster Security for Kubernetes Operator includes the following two custom resources:

1. **Central** - The central resource is a logical grouping of the following services:
    - **Central**: Central is the Red Hat Advanced Cluster Security for Kubernetes application management interface and services. It handles data persistence, API interactions, and user interface (RHACS Portal) access. You can use the same Central instance to secure multiple OpenShift Container Platform or Kubernetes clusters.
    - **Scanner**: Scanner is a Red Hat-developed and certified vulnerability scanner for scanning container images and their associated database. It analyzes all image layers to check known vulnerabilities from the Common Vulnerabilities and Exposures (CVEs) list. Scanner also identifies vulnerabilities in packages installed by package managers and in dependencies for multiple programming languages.
2. **SecuredCluster** - The secured cluster resource is a logical grouping of the following services:
    - **Sensor**: Sensor is the service responsible for analyzing and monitoring the cluster. It handles interactions with the OpenShift Container Platform or Kubernetes API server for policy detection and enforcement, and it coordinates with Collector.
    - **Collector**: Collector analyzes and monitors container activity on cluster nodes. It collects information about container runtime and network activity. It then sends the collected data to Sensor.
    - **Admission Control**: The admission controller prevents users from creating workloads that violate security policies in Red Hat Advanced Cluster Security for Kubernetes.

The following steps represent a high-level workflow for installing Red Hat Advanced Cluster Security for Kubernetes by using an Operator:

1. Install the Red Hat Advanced Cluster Security for Kubernetes Operator from OperatorHub in the cluster where you want to install Central.
2. Configure and deploy the **Central** custom resource.
3. Generate and apply an init bundle. The init bundle contains the secrets that provide linking between Central and the secured clusters.
4. Install the Red Hat Advanced Cluster Security for Kubernetes Operator in all clusters that you want to monitor.

5. Configure and deploy the **SecuredCluster** custom resource in each individual cluster that you want to monitor.

3.1. INSTALLING THE RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES OPERATOR

Using the OperatorHub provided with OpenShift Container Platform is the easiest way to install Red Hat Advanced Cluster Security for Kubernetes.

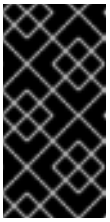
Prerequisites

- You have access to an OpenShift Container Platform cluster using an account with Operator installation permissions.
- You must be using OpenShift Container Platform 4.6 or later.

Procedure

1. Navigate in the web console to the **Operators → OperatorHub** page.
2. If Red Hat Advanced Cluster Security for Kubernetes is not displayed, enter **Advanced Cluster Security** into the **Filter by keyword** box to find the Red Hat Advanced Cluster Security for Kubernetes Operator.
3. Select the **Red Hat Advanced Cluster Security for Kubernetes Operator**to view the details page.
4. Read the information about the Operator and click **Install**.
5. On the **Install Operator** page:
   - Keep the default value for **Installation mode** as **All namespaces on the cluster**.
   - Choose a specific namespace in which to install the Operator for the **Installed namespace** field. Red Hat recommends installing the Red Hat Advanced Cluster Security for Kubernetes Operator in the **rhacs-operator** namespace.
   - Select automatic or manual updates for **Update approval**.
     If you choose automatic updates, when a new version of the Operator is available, Operator Lifecycle Manager (OLM) automatically upgrades the running instance of your Operator.

     If you choose manual updates, when a newer version of the Operator is available, OLM creates an update request. As a cluster administrator, you must then manually approve that update request to update the Operator to the new version.

     IMPORTANT

     If you choose manual updates, you should update the RHACS Operator in all secured clusters when you update the RHACS Operator in the cluster where Central is installed. The secured clusters and the cluster where Central is installed should have the same version to ensure optimal functionality.
6. Click **Install**.

Verification

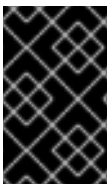
- After the installation completes, navigate to **Operators → Installed Operators** to verify that the Red Hat Advanced Cluster Security for Kubernetes Operator is listed with the status of **Succeeded**.

Next Step

- Install, configure, and deploy the **Central** custom resource.

3.2. INSTALLING CENTRAL

The main component of Red Hat Advanced Cluster Security for Kubernetes is called Central. You can install Central on OpenShift Container Platform by using the **Central** custom resource. You deploy Central only once, and you can monitor multiple separate clusters by using the same Central installation.



IMPORTANT

When you install Red Hat Advanced Cluster Security for Kubernetes for the first time, you must first install the **Central** custom resource because the **SecuredCluster** custom resource installation is dependent on certificates that Central generates.

Prerequisites

- You must be using OpenShift Container Platform 4.6 or later.

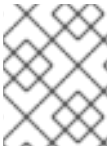
Procedure

1. On the OpenShift Container Platform web console, navigate to the **Operators → Installed Operators** page.
2. Select the Red Hat Advanced Cluster Security for Kubernetes Operator from the list of installed Operators.
3. If you have installed the Operator in the recommended namespace, OpenShift Container Platform lists the project as **rhacs-operator**. Select **Project: rhacs-operator → Create project**.

WARNING

- If you have installed the Operator in a different namespace, OpenShift Container Platform shows the name of that namespace rather than **rhacs-operator**.
- You must install the Red Hat Advanced Cluster Security for Kubernetes **Central** custom resource in its own project and not in the **rhacs-operator** and **openshift-operator** projects, or in the project in which you have installed the Red Hat Advanced Cluster Security for Kubernetes Operator.

4. Enter the new project name (for example, **stackrox**), and click **Create**. Red Hat recommends that you use **stackrox** as the project name.
5. Under the **Provided APIs** section, select **Central**. Click **Create Central**.
6. Enter a name for your **Central** custom resource and add any labels you want to apply. Otherwise, accept the default values for the available options.
7. Click **Create**.



NOTE

If you are using the cluster-wide proxy, Red Hat Advanced Cluster Security for Kubernetes uses that proxy configuration to connect to the external services.

Next Steps

1. Verify Central installation.
2. Optional: Configure Central options.
3. Generate an init bundle.

Additional resources

- Configuring the cluster-wide proxy

3.3. VERIFYING CENTRAL INSTALLATION

After Central finishes installing, log in to the RHACS portal to verify the successful installation of Central.

Procedure

1. On the OpenShift Container Platform web console, navigate to the **Operators → Installed Operators** page.
2. Select the Red Hat Advanced Cluster Security for Kubernetes Operator from the list of installed Operators.
3. Select the **Central** tab.
4. From the **Centrals** list, select **stackrox-central-services** to view its details.
5. To get the password for the **admin** user, you can either:
   - Click the link under **Admin Password Secret Reference**.
   - Use the OpenShift Container Platform CLI to enter the command listed under **Admin Credentials Info**:

     ```
     $ oc -n stackrox get secret central-htpasswd -o go-template='{{index .data "password" | base64decode}}'
     ```
6. Find the link to the RHACS portal by using the OpenShift Container Platform CLI command:

```
$ oc -n stackrox get route central -o jsonpath="{.status.ingress[0].host}"
```

Alternatively, you can use the Red Hat Advanced Cluster Security for Kubernetes web console to find the link to the RHACS portal by performing the following commands:

a. Navigate to **Networking → Routes**.

b. Find the **central** Route and click on the RHACS portal link under the **Location** column.

7. Log in to the RHACS portal using the username **admin** and the password that you retrieved in a previous step. Until Red Hat Advanced Cluster Security for Kubernetes is completely configured (for example, you have the **Central** resource and at least one **SecuredCluster** resource installed and configured), no data is available in the dashboard. The **SecuredCluster** resource can be installed and configured on the same cluster as the **Central** resource. Clusters with the **SecuredCluster** resource are similar to managed clusters in Red Hat Advanced Cluster Management (RHACM).

Next Steps

1. Optional: Configure central settings.
2. Generate an init bundle containing the cluster secrets that allows communication between the **Central** and **SecuredCluster** resources. You need to download this bundle, use it to generate resources on the clusters you want to secure, and securely store it.

3.4. CENTRAL CONFIGURATION OPTIONS

When you create a Central instance, the Operator lists the following configuration options for the **Central** custom resource.

3.4.1. Central settings

Parameter	Description
central.adminPasswordSecret	Specify a secret that contains the administrator password in the **password** data item. If omitted, the operator autogenerates a password and stores it in the **password** item in the **central-htpasswd** secret.
central.defaultTLSSecret	By default, Central only serves an internal TLS certificate, which means that you need to handle TLS termination at the ingress or load balancer level. If you want to terminate TLS in Central and serve a custom server certificate, you can specify a secret containing the certificate and private key.
central.adminPasswordGenerationDisabled	Set this parameter to **true** to disable the automatic administrator password generation. Use this only after you perform the first-time setup of alternative authentication methods. Do not use this for initial installation. Otherwise, you must reinstall the custom resource to log back in.
central.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Central. This parameter is mainly used for infrastructure nodes.

Parameter	Description
central.exposure.loadBalancer.enabled	Set this to **true** to expose Central through a load balancer.
central.exposure.loadBalancer.port	Use this parameter to specify a custom port for your load balancer.
central.exposure.loadBalancer.ip	Use this parameter to specify a static IP address reserved for your load balancer.
central.exposure.route.enabled	Set this to **true** to expose Central through an OpenShift route. The default value is **false**.
central.exposure.nodeport.enabled	Set this to **true** to expose Central through a node port. The default value is **false**.
central.exposure.nodeport.port	Use this to specify an explicit node port.
central.nodeSelector	If you want this component to only run on specific nodes, you can configure a node selector by using this parameter.
central.persistence.hostPath.path	Specify a host path to store persistent data in a directory on the host. Red Hat does not recommend using this. If you need to use host path, you must use it with a node selector.
central.persistence.persistentVolumeClaim.claimName	The name of the PVC to manage persistent data. If no PVC with the given name exists, it will be created. The default value is **stackrox-db** if not set. To prevent data losses the PVC is not removed automatically with Central`s deletion.
central.persistence.persistentVolumeClaim.size	The size of the persistent volume when created through the claim. This is automatically generated by default.
central.persistence.persistentVolumeClaim.storageClassName	The name of the storage class to use for the PVC. If your cluster is not configured with a default storage class, you must provide a value for this parameter.
central.resources.limits	Use this parameter to override the default resource limits for the Central.
central.resources.requests	Use this parameter to override the default resource requests for the Central.
central.imagePullSecrets	Use this parameter to specify the image pull secrets for the Central image.

3.4.2. Scanner settings

Parameter	Description
scanner.analyzer.nodeSelector	If you want this scanner to only run on specific nodes, you can configure a node selector by using this parameter.
scanner.analyzer.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner. This parameter is mainly used for infrastructure nodes.
scanner.analyzer.resources.limits	Use this parameter to override the default resource limits for the scanner.
scanner.analyzer.resources.requests	Use this parameter to override the default resource requests for the scanner.
scanner.analyzer.scaling.autoScaling	When enabled, the number of analyzer replicas is managed dynamically based on the load, within the limits specified.
scanner.analyzer.scaling.maxReplicas	Specifies the maximum replicas to be used the analyzer autoscaling configuration
scanner.analyzer.scaling.minReplicas	Specifies the minimum replicas to be used the analyzer autoscaling configuration
scanner.analyzer.scaling.replicas	When autoscaling is disabled, the number of replicas will always be configured to match this value.
scanner.db.nodeSelector	If you want this component to only run on specific nodes, you can configure a node selector by using this parameter.
scanner.db.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner DB. This parameter is mainly used for infrastructure nodes.
scanner.db.resources.limits	Use this parameter to override the default resource limits for the scanner.
scanner.db.resources.requests	Use this parameter to override the default resource requests for the scanner.
scanner.scannerComponent	If you do not want to deploy Scanner, you can disable it by using this parameter. If you disable Scanner, all other settings in this section have no effect. Red Hat does not recommend disabling Red Hat Advanced Cluster Security for Kubernetes Scanner.

3.4.3. General and miscellaneous settings

Parameter	Description
tls.additionalCAs	Additional Trusted CA certificates for the secured cluster to trust. This is typically used when integrating with services using a private certificate authority.
misc.createSCCs	Specify **true** to create **SecurityContextConstraints** (SCCs) for Central. It might cause issues in some environments.

3.5. GENERATING AN INIT BUNDLE

Before you install the **SecuredCluster** resource on a cluster, you must create an init bundle. The cluster that has **SecuredCluster** installed and configured then uses this bundle to authenticate with Central.

You can create an init bundle by using the RHACS portal (recommended) or by using the roxctl CLI.

3.5.1. Generating an init bundle by using the RHACS portal

You can create an init bundle containing secrets by using the RHACS portal.

Procedure

1. Find the address of the RHACS portal based on your exposure method:
   a. For a route:

      ```
      $ oc get route central -n stackrox
      ```

   b. For a load balancer:

      ```
      $ oc get service central-loadbalancer -n stackrox
      ```

   c. For port forward:
      i. Run the following command:

         ```
         $ oc port-forward svc/central 18443:443 -n stackrox
         ```

      ii. Navigate to **https://localhost:18443/**.
2. On the RHACS portal, navigate to **Platform Configuration → Integrations**.
3. Navigate to the **Authentication Tokens** section and click on **Cluster Init Bundle**.
4. Click **Generate bundle**.
5. Enter a name for the cluster init bundle and click **Generate**.
6. Click **Download Kubernetes Secret File** to download the generated bundle.

IMPORTANT

Store this bundle securely because it contains secrets. You can use the same bundle to create multiple secured clusters.

Next Step

1. Use the OpenShift Container Platform CLI to create resources using the init bundle.
2. Install Red Hat Advanced Cluster Security for Kubernetes in all clusters that you want to monitor.

3.5.2. Generating an init bundle by using the roxctl CLI

You can create an init bundle with secrets by using the **roxctl** CLI.

Prerequisites

You have configured the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables.

- Set the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables:

```
$ export ROX_API_TOKEN=<api_token>
```

```
$ export ROX_CENTRAL_ADDRESS=<address>:<port_number>
```

Procedure

- Run the following command to generate a cluster init bundle containing secrets:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output-secrets cluster_init_bundle.yaml
```

IMPORTANT

Make sure that you store this bundle securely because it contains secrets. You can use the same bundle to set up multiple secured clusters.

3.5.3. Additional resources

- Installing the **roxctl** CLI
- Using the **roxctl** CLI

3.6. CREATING RESOURCES BY USING THE INIT BUNDLE

Before you install secured clusters, you must use the init bundle to create the required resources on the cluster that will allow the services on the secured clusters to communicate with Central.

Prerequisites

- You must have generated an init bundle containing secrets.

Procedure

- Using the OpenShift Container Platform CLI, run the following command to create the resources:

```
$ oc create -f <init_bundle>.yaml \ 1
  -n <stackrox> 2
```

1. Specify the file name of the init bundle containing the secrets.
2. Specify the name of the project where you installed Central.

Next Step

- Install Red Hat Advanced Cluster Security for Kubernetes in all clusters that you want to monitor.

3.7. INSTALLING SECURED CLUSTER SERVICES

You can install secured cluster services on your clusters by using the **SecuredCluster** custom resource. You must install the secured cluster services on every cluster in your environment that you want to monitor.

CAUTION

To install Collector on systems that have Unified Extensible Firmware Interface (UEFI) and that have Secure Boot enabled, you must use eBPF probes because kernel modules are unsigned, and the UEFI firmware cannot load unsigned packages. Collector identifies Secure Boot status at the start and switches to eBPF probes if required.

Prerequisites

- You must be using OpenShift Container Platform 4.6 or later.
- You must have generated an init bundle and already created the required resources by using the init bundle.

Procedure

1. On the OpenShift Container Platform web console, navigate to the **Operators → Installed Operators** page.
2. Select the Red Hat Advanced Cluster Security for Kubernetes Operator from the list of installed operators.
3. By default, OpenShift Container Platform lists the project as **rhacs-operator**. Select **Project: rhacs-operator → Create project**.

WARNING

You must install Red Hat Advanced Cluster Security for Kubernetes **SecuredCluster** resource in its own project and not the default **openshift-operators** project.

4. Enter the new project name as **stackrox** or some other name, and click **Create**.
5. Under the **Provided APIs** section, select **Secured Cluster**.
6. Choose **Create SecuredCluster**.
7. Enter a name for your **SecuredCluster** custom resource.
8. For **Central Endpoint**, enter the address and port number of your Central instance. For example, if Central is available at **https://central.example.com**, then specify the central endpoint as **central.example.com:443**. The default value of **central.stackrox.svc:443** only works when you install secured cluster services and Central in the same cluster.
9. Accept the default values or configure custom values for the available options.
10. Click **Create**.

Next step

1. Optional: Configure additional secured cluster settings.
2. Verify Red Hat Advanced Cluster Security for Kubernetes installation.

3.8. SECURED CLUSTER CONFIGURATION OPTIONS

When you create a Central instance, the Operator lists the following configuration options for the **Central** custom resource.

3.8.1. Required Configuration Settings

Parameter	Description
centralEndpoint	The endpoint of Central instance to connect to, including the port number. If using a non-gRPC capable load balancer, use the WebSocket protocol by prefixing the endpoint address with **wss://**. If you do not specify a value for this paramter, Sensor attempts to connect to a Central instance running in the same namespace.
clusterName	The unique name of this cluster, which shows up in the RHACS portal. After the name is set by using this parameter, you cannot change it again. To change the name, you must delete and recreate the object.

3.8.2. Admission controller settings

Parameter	Description
admissionControl.listenOnCreates	Specify **true** to enable preventive policy enforcement for object creations. The default value is **false**.
admissionControl.listenOnEvents	Specify **true** to enable monitoring and enforcement for Kubernetes events, such as **port-forward** and **exec** events. It is used to control access to resources through the Kubernetes API. The default value is **true**.
admissionControl.listenOnUpdates	Specify **true** to enable preventive policy enforcement for object updates. It will not have any effect unless **Listen On Creates** is set to **true** as well. The default value is **false**.
admissionControl.nodeSelector	If you want this component to only run on specific nodes, you can configure a node selector using this parameter.
admissionControl.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Admission Control. This parameter is mainly used for infrastructure nodes.
admissionControl.resources.limits	Use this parameter to override the default resource limits for the admission controller.
admissionControl.resources.requests	Use this parameter to override the default resource requests for the admission controller.
admissionControl.bypass	Use one of the following values to configure the bypassing of admission controller enforcement: • **BreakGlassAnnotation** to enable bypassing the admission controller via the **admission.stackrox.io/break-glass** annotation. • **Disabled** to disable the ability to bypass admission controller enforcement for the secured cluster. The default value is **BreakGlassAnnotation**.
admissionControl.contactImageScanners	Use one of the following values to specify if the admission controller must connect to the image scanner: • **ScanIfMissing** if the scan results for the image are missing. • **DoNotScanInline** to skip scanning the image when processing the admission request. The default value is **DoNotScanInline**.
admissionControl.timeoutSeconds	Use this parameter to specify the maximum number of seconds Red Hat Advanced Cluster Security for Kubernetes must wait for an admission review before marking it as fail open.

3.8.3. Scanner configuration

Use Scanner configuration settings to modify the local cluster scanner for the OpenShift Container Registry (OCR).

Parameter	Description
scanner.analyzer.nodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner to only schedule on nodes with the specified label.
scanner.analyzer.resources.requests.memory	The memory request for the Scanner container. Use this parameter to override the default value.
scanner.analyzer.resources.requests.cpu	The CPU request for the Scanner container. Use this parameter to override the default value.
scanner.analyzer.resources.limits.memory	The memory limit for the Scanner container. Use this parameter to override the default value.
scanner.analyzer.resources.limits.cpu	The CPU limit for the Scanner container. Use this parameter to override the default value.
scanner.scaling.autoscaling	If you set this option to **Disabled**, Red Hat Advanced Cluster Security for Kubernetes disables autoscaling on the Scanner deployment. The default value is **Enabled**.
scanner.scaling.minReplicas	The minimum number of replicas for autoscaling. The default value is **2**.
scanner.scaling.maxReplicas	The maximum number of replicas for autoscaling. The default value is **5**.
scanner.scaling.replicas	The default number of replicas. The default value is **3**.
scanner.Tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner.
scanner.db.nodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner DB to only schedule on nodes with the specified label.
scanner.db.resources.requests.memory	The memory request for the Scanner DB container. Use this parameter to override the default value.
scanner.db.resources.requests.cpu	The CPU request for the Scanner DB container. Use this parameter to override the default value.

Parameter	Description
scanner.db.resources.limits.memory	The memory limit for the Scanner DB container. Use this parameter to override the default value.
scanner.db.resources.limits.cpu	The CPU limit for the Scanner DB container. Use this parameter to override the default value.
scanner.db.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner DB.
scanner.scannerComponent	If you set this option to **Disabled**, Red Hat Advanced Cluster Security for Kubernetes does not deploy the Scanner deployment. Do not disable the Scanner on OpenShift Container Platform clusters. The default value is **AutoSense**.

3.8.4. Image configuration

Use image configuration settings when you are using a custom registry.

Parameter	Description
imagePullSecrets.name	Additional image pull secrets to be taken into account for pulling images.

3.8.5. Per node settings

Per node settings define the configuration settings for components that run on each node in a cluster to secure the cluster. These components are Collector and Compliance.

Parameter	Description
perNode.collector.collection	The method for system-level data collection. The default value is **KernelModule**. Red Hat recommend using **KernelModule** as the value for this parameter. If you select **NoCollection**, you will not be able to see any information about network activity and process executions. Options include **NoCollection**, **EBPF**, and **KernelModule**.
perNode.collector.imageFlavor	The image type to use for Collector. You can specify it as **Regular** or **Slim**. **Regular** images are bigger in size, but contain kernel modules for most kernels. If you use the **Slim** image type, you must ensure that your Central instance is connected to the internet, or regularly receives Collector support package updates. The default value is **Slim**.
perNode.collector.resources.limits	Use this parameter to override the default resource limits for Collector.
perNode.collector.resources.requests	Use this parameter to override the default resource requests for Collector.

Parameter	Description
perNode.compliance.resources.requests	Use this parameter to override the default resource requests for Compliance.
perNode.compliance.resources.limits	Use this parameter to override the default resource limits for Compliance.

3.8.6. Taint Tolerations settings

Parameter	Description
taintToleration	To ensure comprehensive monitoring of your cluster activity, Red Hat Advanced Cluster Security for Kubernetes runs services on every node in the cluster, including tainted nodes by default. If you do not want this behavior, specify **AvoidTaints** for this parameter.

3.8.7. Sensor configuration

This configuration defines the settings of the Sensor components, which runs on one node in a cluster.

Parameter	Description
sensor.nodeSelector	If you want Sensor to only run on specific nodes, you can configure a node selector.
sensor.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Sensor. This parameter is mainly used for infrastructure nodes.
sensor.resources.limits	Use this parameter to override the default resource limits for Sensor.
sensor.resources.requests	Use this parameter to override the default resource requests for Sensor.

3.8.8. General and miscellaneous settings

Parameter	Description
tls.additionalCAs	Additional trusted CA certificates for the secured cluster. These certificates are used when integrating with services using a private certificate authority.
misc.createSCCs	Set this to **true** to create SCCs for Central. It may cause issues in some environments.

Parameter	Description
customize.annotations	Allows specifying custom annotations for the Central deployment.
customize.envVars	Advanced settings to configure environment variables.
egress.connectivityPolicy	Configures whether Red Hat Advanced Cluster Security for Kubernetes should run in online or offline mode. In offline mode, automatic updates of vulnerability definitions and kernel modules are disabled.

3.9. VERIFYING INSTALLATION

After you complete the installation, run a few vulnerable applications and navigate to the RHACS portal to evaluate the results of security assessments and policy violations.

NOTE

The sample applications listed in the following section contain critical vulnerabilities and they are specifically designed to verify the build and deploy-time assessment features of Red Hat Advanced Cluster Security for Kubernetes.

To verify installation:

1. Find the address of the RHACS portal based on your exposure method:
   a. For a route:

      ```
      $ oc get route central -n stackrox
      ```

   b. For a load balancer:

      ```
      $ oc get service central-loadbalancer -n stackrox
      ```

   c. For port forward:
      i. Run the following command:

         ```
         $ oc port-forward svc/central 18443:443 -n stackrox
         ```

      ii. Navigate to **https://localhost:18443/**.
2. Using the OpenShift Container Platform CLI, create a new project:

   ```
   $ oc new-project test
   ```

3. Start some applications with critical vulnerabilities:

   ```
   $ oc run shell --labels=app=shellshock,team=test-team \
     --image=vulnerables/cve-2014-6271 -n test
   ```

```
$ oc run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes automatically scans these deployments for security risk and policy violations as soon as they are submitted to the cluster. Navigate to the RHACS portal to view the violations. You can log in to the RHACS portal by using the default username **admin** and the generated password.

3.10. ADDING A NEW CLUSTER TO RHACS

To add more clusters to Red Hat Advanced Cluster Security for Kubernetes, you must install the Red Hat Advanced Cluster Security for Kubernetes Operator in every cluster that you want to add.

The following steps represent the high-level flow for adding additional clusters to Red Hat Advanced Cluster Security for Kubernetes:

1. Install the Red Hat Advanced Cluster Security for Kubernetes Operator in your cluster.
2. Use an existing init bundle or generate a new init bundle.
3. Create resources in your cluster by using the init bundle.
4. Install secured cluster services on your cluster.

CHAPTER 4. INSTALLING USING HELM CHARTS

4.1. INSTALLING QUICKLY USING HELM CHARTS

Red Hat Advanced Cluster Security for Kubernetes installs a set of services on your OpenShift Container Platform cluster. This topic describes the installation procedure for installing Red Hat Advanced Cluster Security for Kubernetes on your OpenShift Container Platform cluster without any customizations.

The following steps represent the high-level installation flow for quickly installing Red Hat Advanced Cluster Security for Kubernetes:

1. Add the Red Hat Advanced Cluster Security for Kubernetes Helm chart repository.
2. Install the **central-services** Helm chart to install the centralized components (Central and Scanner).
3. Generate an init bundle.
4. Install the **secured-cluster-services** Helm chart to install the per-cluster and per-node components (Sensor, Admission Controller, and Collector).

Before you install:

- Understand Red Hat Advanced Cluster Security for Kubernetes architecture .
- Review the prerequisites for installing Red Hat Advanced Cluster Security for Kubernetes .

4.1.1. Adding the Helm chart repository

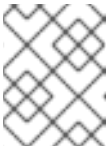
Procedure

- Add Red Hat Advanced Cluster Security for Kubernetes charts repository.

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

  The Helm repository for Red Hat Advanced Cluster Security for Kubernetes includes two Helm charts for installing different components.

  - Central services Helm chart (**central-services**) for installing the centralized components (Central and Scanner).

    NOTE

    You deploy centralized components only once and you can monitor multiple separate clusters by using the same installation.

  - Secured Cluster Services Helm chart (**secured-cluster-services**) for installing the per-cluster (Sensor and Admission controller) and per-node (Collector) components.

NOTE

Deploy the per-cluster components into each cluster that you want to monitor and deploy the per-node components in all nodes that you want to monitor.

Verification

- Run the following command to verify the added chart repository:

```
$ helm search repo -l rhacs/
```

4.1.2. Installing the central-services Helm chart without customization

Use the following instructions to install the **central-services** Helm chart to deploy the centralized components (Central and Scanner).

Procedure

- Run the following command to install Central services and expose Central using a route:

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.allowNone=true \
  --set central.exposure.route.enabled=true
```

- Or, run the following command to install Central services and expose Central using a load balancer:

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.allowNone=true \
  --set central.exposure.loadBalancer.enabled=true
```

- Or, run the following command to install Central services and expose Central using port forward:

```
$ helm install -n stackrox \
  --create-namespace stackrox-central-services rhacs/central-services \
  --set imagePullSecrets.allowNone=true
```

IMPORTANT

If you are installing Red Hat Advanced Cluster Security for Kubernetes in a cluster that requires a proxy to connect to external services, you must specify your proxy configuration by using the **proxyConfig** parameter. For example:

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

The output of the installation command includes:

- An automatically generated administrator password.
- Instructions on storing all the configuration values.
- Any warnings that Helm generates.

4.1.3. Generating an init bundle

Before you install the **SecuredCluster** resource on a cluster, you must create an init bundle. The cluster that has **SecuredCluster** installed and configured then uses this bundle to authenticate with Central.

4.1.3.1. Generating an init bundle by using the roxctl CLI

You can create an init bundle with secrets by using the **roxctl** CLI.

Prerequisites

You have configured the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables.

- Set the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables:

```
$ export ROX_API_TOKEN=<api_token>
```

```
$ export ROX_CENTRAL_ADDRESS=<address>:<port_number>
```

Procedure

- Run the following command to generate a cluster init bundle containing secrets:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output cluster_init_bundle.yaml
```

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output-secrets cluster_init_bundle.yaml
```

IMPORTANT

Make sure that you store this bundle securely because it contains secrets. You can use the same bundle to set up multiple secured clusters.

Additional resources

- Installing the **roxctl** CLI
- Generating an init bundle by using the RHACS portal

4.1.4. Installing the secured-cluster-services Helm chart without customization

Use the following instructions to install the **secured-cluster-services** Helm chart to deploy the per-cluster and per-node components (Sensor, Admission Controller, and Collector).

CAUTION

To install Collector on systems that have Unified Extensible Firmware Interface (UEFI) and that have Secure Boot enabled, you must use eBPF probes because kernel modules are unsigned, and the UEFI firmware cannot load unsigned packages. Collector identifies Secure Boot status at the start and switches to eBPF probes if required.

Prerequisites

- You must have the address and the port number that you are exposing the Central service on.

Procedure

- Run the following command on other Kubernetes based clusters:

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <path_to_cluster_init_bundle.yaml> \ 1
  --set clusterName=<name_of_the_secured_cluster> \
  --set centralEndpoint=<endpoint_of_central_service> 2
```

1. Use the **-f** option to specify the path for the init bundle.
2. Specify the address and port number for Central. For example, **acs.domain.com:443**.

- Run the following command on OpenShift Container Platform clusters:

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <path_to_cluster_init_bundle.yaml> \ 1
  --set clusterName=<name_of_the_secured_cluster> \
  --set centralEndpoint=<endpoint_of_central_service> 2
  --set scanner.disable=false
```

1. Use the **-f** option to specify the path for the init bundle.
2. Specify the address and port number for Central. For example, **acs.domain.com:443**.

4.1.5. Verifying installation

After you complete the installation, run a few vulnerable applications and navigate to the RHACS portal to evaluate the results of security assessments and policy violations.

NOTE

The sample applications listed in the following section contain critical vulnerabilities and they are specifically designed to verify the build and deploy-time assessment features of Red Hat Advanced Cluster Security for Kubernetes.

To verify installation:

1. Find the address of the RHACS portal based on your exposure method:
   a. For a route:

```
$ oc get route central -n stackrox
```

   b. For a load balancer:

```
$ oc get service central-loadbalancer -n stackrox
```

   c. For port forward:
      i. Run the following command:

```
$ oc port-forward svc/central 18443:443 -n stackrox
```

      ii. Navigate to **https://localhost:18443**/.
2. Using the OpenShift Container Platform CLI, create a new project:

```
$ oc new-project test
```

3. Start some applications with critical vulnerabilities:

```
$ oc run shell --labels=app=shellshock,team=test-team \
  --image=vulnerables/cve-2014-6271 -n test
$ oc run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes automatically scans these deployments for security risk and policy violations as soon as they are submitted to the cluster. Navigate to the RHACS portal to view the violations. You can log in to the RHACS portal by using the default username **admin** and the generated password.

4.1.6. Additional resources

- Installing with customizations using Helm charts

4.2. INSTALLING WITH CUSTOMIZATIONS USING HELM CHARTS

High-level installation flow:

1. Add Red Hat Advanced Cluster Security for Kubernetes Helm chart repository.
2. Configure the **central-services** Helm chart.
3. Install the **central-services** Helm chart to install the centralized components (Central and Scanner).
4. Generate an init bundle.
5. Configure the **secured-cluster-services** Helm chart.

6. Install the **secured-cluster-services** Helm chart to install the per-cluster and per-node components (Sensor, Admission Controller, and Collector).

Before you install:

- Understand Red Hat Advanced Cluster Security for Kubernetes architecture .
- Review the prerequisites for installing Red Hat Advanced Cluster Security for Kubernetes .

4.2.1. Adding the Helm chart repository

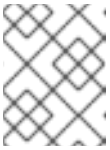
Procedure

- Add Red Hat Advanced Cluster Security for Kubernetes charts repository.

```
$ helm repo add rhacs https://mirror.openshift.com/pub/rhacs/charts/
```

The Helm repository for Red Hat Advanced Cluster Security for Kubernetes includes two Helm charts for installing different components.

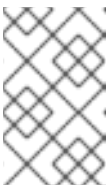
  - Central services Helm chart (**central-services**) for installing the centralized components (Central and Scanner).



NOTE

You deploy centralized components only once and you can monitor multiple separate clusters by using the same installation.

  - Secured Cluster Services Helm chart (**secured-cluster-services**) for installing the per-cluster (Sensor and Admission controller) and per-node (Collector) components.



NOTE

Deploy the per-cluster components into each cluster that you want to monitor and deploy the per-node components in all nodes that you want to monitor.

Verification

- Run the following command to verify the added chart repository:

```
$ helm search repo -l rhacs/
```

4.2.2. Configuring the central-services Helm chart

This section describes Helm chart configuration parameters that you can use with the **helm install** and **helm upgrade** commands. You can specify these parameters by using the **--set** option or by creating YAML configuration files.

Create the following files for configuring the Helm chart for installing Red Hat Advanced Cluster Security for Kubernetes:

- Public configuration file **values-public.yaml**: Use this file to save all non-sensitive configuration options.
- Private configuration file **values-private.yaml**: Use this file to save all sensitive configuration options. Make sure that you store this file securely.

4.2.2.1. Private configuration file

This section lists the configurable parameters of the **values-private.yaml** file. There are no default values for these parameters.

4.2.2.1.1. Image pull secrets

The credentials that are required for pulling images from the registry depend on the following factors:

- If you are using a custom registry, you must specify these parameters:
  - **imagePullSecrets.username**
  - **imagePullSecrets.password**
  - **image.registry**
- If you do not use a username and password to log in to the custom registry, you must specify one of the following parameters:
  - **imagePullSecrets.allowNone**
  - **imagePullSecrets.useExisting**
  - **imagePullSecrets.useFromDefaultServiceAccount**

Parameter	Description
imagePullSecrets.username	The username of the account that is used to log in to the registry.
imagePullSecrets.password	The password of the account that is used to log in to the registry.
imagePullSecrets.allowNone	Use **true** if you are using a custom registry and it allows pulling images without credentials.
imagePullSecrets.useExisting	A comma-separated list of secrets as values. For example, **secret1, secret2, secretN**. Use this option if you have already created pre-existing image pull secrets with the given name in the target namespace.
imagePullSecrets.useFromDefaultServiceAccount	Use **true** if you have already configured the default service account in the target namespace with sufficiently scoped image pull secrets.

4.2.2.1.2. Proxy configuration

If you are installing Red Hat Advanced Cluster Security for Kubernetes in a cluster that requires a proxy to connect to external services, you must specify your proxy configuration by using the **proxyConfig** parameter. For example:

```
env:
  proxyConfig: |
    url: http://proxy.name:port
    username: username
    password: password
    excludes:
    - some.domain
```

Parameter	Description
env.proxyConfig	Your proxy configuration.

4.2.2.1.3. Central

Configurable parameters for Central.

For a new installation, you can skip the following parameters:

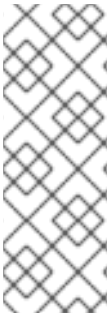
- **central.jwtSigner.key**
- **central.serviceTLS.cert**
- **central.serviceTLS.key**
- **central.adminPassword.value**
- **central.adminPassword.htpasswd**
- When you do not specify values for these parameters the Helm chart autogenerates values for them.
- If you want to modify these values you can use the **helm upgrade** command and specify the values using the **--set** option.

IMPORTANT

For setting the administrator password, you can only use either **central.adminPassword.value** or **central.adminPassword.htpasswd**, but not both.

Parameter	Description
central.jwtSigner.key	A private key which Red Hat Advanced Cluster Security for Kubernetes should use for signing JSON web tokens (JWTs) for authentication.

Parameter	Description
central.serviceTLS.cert	An internal certificate that the Central service should use for deploying Central.
central.serviceTLS.key	The private key of the internal certificate that the Central service should use.
central.defaultTLS.cert	The user-facing certificate that Central should use. Red Hat Advanced Cluster Security for Kubernetes uses this certificate for RHACS portal. ● For a new installation, you must provide a certificate, otherwise, Red Hat Advanced Cluster Security for Kubernetes installs Central by using a self-signed certificate. ● If you are upgrading, Red Hat Advanced Cluster Security for Kubernetes uses the existing certificate and its key.
central.defaultTLS.key	The private key of the user-facing certificate that Central should use. ● For a new installation, you must provide the private key, otherwise, Red Hat Advanced Cluster Security for Kubernetes installs Central by using a self-signed certificate. ● If you are upgrading, Red Hat Advanced Cluster Security for Kubernetes uses the existing certificate and its key.
central.adminPassword.value	Administrator password for logging into Red Hat Advanced Cluster Security for Kubernetes.
central.adminPassword.htpasswd	Administrator password for logging into Red Hat Advanced Cluster Security for Kubernetes. This password is stored in hashed format using bcrypt.



NOTE

If you are using **central.adminPassword.htpasswd** parameter, you must use a bcrypt encoded password hash. You can run the command **htpasswd -nB admin** to generate a password hash. For example,

```
htpasswd: |
  admin:<bcrypt-hash>
```

4.2.2.1.4. Scanner

Configurable parameters for Scanner.

For a new installation, you can skip the following parameters and the Helm chart autogenerates values for them. Otherwise, if you are upgrading to a new version, specify the values for the following parameters:

- **scanner.dbPassword.value**
- **scanner.serviceTLS.cert**
- **scanner.serviceTLS.key**
- **scanner.dbServiceTLS.cert**
- **scanner.dbServiceTLS.key**

Parameter	Description
scanner.dbPassword.value	The password to use for authentication with Scanner database. Do not modify this parameter because Red Hat Advanced Cluster Security for Kubernetes automatically creates and uses its value internally.
scanner.serviceTLS.cert	An internal certificate that the Scanner service should use for deploying Scanner.
scanner.serviceTLS.key	The private key of the internal certificate that the Scanner service should use.
scanner.dbServiceTLS.cert	An internal certificate that the Scanner-db service should use for deploying Scanner database.
scanner.dbServiceTLS.key	The private key of the internal certificate that the Scanner-db service should use.

4.2.2.2. Public configuration file

This section lists the configurable parameters of the **values-public.yaml** file.

4.2.2.2.1. Image pull secrets

Image pull secrets are the credentials required for pulling images from your registry.

Parameter	Description
imagePullSecrets.allowNone	Use **true** if you are using a custom registry and it allows pulling images without credentials.
imagePullSecrets.useExisting	A comma-seprated list of secrets as values. For example, **secret1, secret2**. Use this option if you have already created pre-existing image pull secrets with the given name in the target namespace.

Parameter	Description
imagePullSecrets.useFromDefaultServiceAccount	Use **true** if you have already configured the default service account in the target namespace with sufficiently scoped image pull secrets.

4.2.2.2.2. Image

Image declares the configuration to set up the main registry, which the Helm chart uses to resolve images for the **central.image**, **scanner.image**, and **scanner.dbImage** parameters.

Parameter	Description
image.registry	Address of your image registry. Either use a hostname, such as **registry.redhat.io**, or a remote registry hostname, such as **us.gcr.io/stackrox-mirror**.

4.2.2.2.3. Environment variables

Red Hat Advanced Cluster Security for Kubernetes automatically detects your cluster environment and sets values for **env.openshift**, **env.istio**, and **env.platform**. Only set these values to override the automatic cluster environment detection.

Parameter	Description
env.openshift	Use **true** for installing on an OpenShift Container Platform cluster and overriding automatic cluster environment detection.
env.istio	Use **true** for installing on an Istio enabled cluster and overriding automatic cluster environment detection.
env.platform	The platform on which you are installing Red Hat Advanced Cluster Security for Kubernetes. Set its value to **default** or **gke** to specify cluster platform and override automatic cluster environment detection.
env.offlineMode	Use **true** to use Red Hat Advanced Cluster Security for Kubernetes in offline mode.

4.2.2.2.4. Additional trusted certificate authorities

The Red Hat Advanced Cluster Security for Kubernetes automatically references the system root certificates to trust. When Central or Scanner must reach out to services that use certificates issued by an authority in your organization or a globally trusted partner organization, you can add trust for these services by specifying the root certificate authority to trust by using the following parameter:

Parameter	Description
additionalCAs.<certificate_name>	Specify the PEM encoded certificate of the root certificate authority to trust.

4.2.2.2.5. Central

Configurable parameters for Central.

- You must specify a persistent storage option as either **hostPath** or **persistentVolumeClaim**.
- For exposing Central deployment for external access. You must specify one parameter, either **central.exposure.loadBalancer**, **central.exposure.nodePort**, or **central.exposure.route**. When you do not specify any value for these parameters, you must manually expose Central or access it by using port-forwarding.

Parameter	Description
central.disableTelemetry	Use **true** to disable online telemetry data collection.
central.endpointsConfig	The endpoint configuration options for Central.
central.nodeSelector	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Central. This parameter is mainly used for infrastructure nodes.
central.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Central. This parameter is mainly used for infrastructure nodes.
central.exposeMonitoring	Specify **true** to expose Prometheus metrics endpoint for Central on port number **9090**.
central.image.registry	A custom registry that overrides the global **image.registry** parameter for the Central image.
central.image.name	The custom image name that overrides the default Central image name (**main**).
central.image.tag	The custom image tag that overrides the default tag for Central image. If you specify you own image tag during a new installation, you must manually increment this tag when you to upgrade to a new version by running the **helm upgrade** command. If you mirror Central images in your own registry, do not modify the original image tags.

Parameter	Description
central.image.fullRef	Full reference including registry address, image name, and image tag for the Central image. Setting a value for this parameter overrides the **central.image.registry**, **central.image.name**, and **central.image.tag** parameters.
central.resources.requests.memory	The memory request for Central to override the default value.
central.resources.requests.cpu	The CPU request for Central to override the default value.
central.resources.limits.memory	The memory limit for Central to override the default value.
central.resources.limits.cpu	The CPU limit for Central to override the default value.
central.persistence.hostPath	The path on the node where Red Hat Advanced Cluster Security for Kubernetes should create a database volume. Red Hat does not recommend using this option.
central.persistence.persistentVolumeClaim.claimName	The name of the persistent volume claim (PVC) you are using.
central.persistence.persistentVolumeClaim.createClaim	Use **true** to create a new persistent volume claim, or **false** to use an existing claim.
central.persistence.persistentVolumeClaim.size	The size (in GiB) of the persistent volume managed by the specified claim.
central.exposure.loadBalancer.enabled	Use **true** to expose Central by using a load balancer.
central.exposure.loadBalancer.port	The port number on which to expose Central. The default port number is 443.
central.exposure.nodePort.enabled	Use **true** to expose Central by using the node port service.
central.exposure.nodePort.port	The port number on which to expose Central. When you skip this parameter, OpenShift Container Platform automatically assigns a port number. Red Hat recommends that you do not specify a port number if you are exposing Red Hat Advanced Cluster Security for Kubernetes by using a node port.

Parameter	Description
central.exposure.route.enabled	Use **true** to expose Central by using a route. This parameter is only available for OpenShift Container Platform clusters.

4.2.2.2.6. Scanner

Configurable parameters for Scanner.

Parameter	Description
scanner.disable	Use **true** to install Red Hat Advanced Cluster Security for Kubernetes without Scanner. When you use it with the **helm upgrade** command, Helm removes existing Scanner deployment.
scanner.replicas	The number of replicas to create for the Scanner deployment. When you use it with the **scanner.autoscaling** parameter, this value sets the initial number of replicas.
scanner.logLevel	Configure the log level for Scanner. Red Hat recommends that you not change the log level's default value (**INFO**).
scanner.nodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner to only schedule on nodes with the specified label.
scanner.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner. This parameter is mainly used for infrastructure nodes.
scanner.autoscaling.disable	Use **true** to disable autoscaling for Scanner deployment. When you disable autoscaling, the **minReplicas** and **maxReplicas** parameters do not have any effect.
scanner.autoscaling.minReplicas	The minimum number of replicas for autoscaling.
scanner.autoscaling.maxReplicas	The maximum number of replicas for autoscaling.
scanner.resources.requests.memory	The memory request for Scanner to override the default value.
scanner.resources.requests.cpu	The CPU request for Scanner to override the default value.

Parameter	Description
scanner.resources.limits.memory	The memory limit for Scanner to override the default value.
scanner.resources.limits.cpu	The CPU limit for Scanner to override the default value.
scanner.dbResources.requests.memory	The memory request for Scanner database deployment to override the default values.
scanner.dbResources.requests.cpu	The CPU request for Scanner database deployment to override the default values.
scanner.dbResources.limits.memory	The memory limit for Scanner database deployment to override the default values.
scanner.dbResources.limits.cpu	The CPU limit for Scanner database deployment to override the default values.
scanner.image.registry	A custom registry for the Scanner image.
scanner.image.name	The custom image name that overrides the default Scanner image name (**scanner**).
scanner.dbImage.registry	A custom registry for the Scanner DB image.
scanner.dbImage.name	The custom image name that overrides the default Scanner DB image name (**scanner-db**).
scanner.dbNodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner DB to only schedule on nodes with the specified label.
scanner.dbTolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner DB. This parameter is mainly used for infrastructure nodes.

4.2.2.2.7. Customization

Use these parameters to specify additional attributes for all objects that Red Hat Advanced Cluster Security for Kubernetes creates.

Parameter	Description
customize.labels	A custom label to attach to all objects.

Parameter	Description
customize.annotations	A custom annotation to attach to all objects.
customize.podLabels	A custom label to attach to all deployments.
customize.podAnnotations	A custom annotation to attach to all deployments.
customize.envVars	A custom environment variable for all containers in all objects.
customize.central.labels	A custom label to attach to all objects that Central creates.
customize.central.annotations	A custom annotation to attach to all objects that Central creates.
customize.central.podLabels	A custom label to attach to all Central deployments.
customize.central.podAnnotations	A custom annotation to attach to all Central deployments.
customize.central.envVars	A custom environment variable for all Central containers.
customize.scanner.labels	A custom label to attach to all objects that Scanner creates.
customize.scanner.annotations	A custom annotation to attach to all objects that Scanner creates.
customize.scanner.podLabels	A custom label to attach to all Scanner deployments.
customize.scanner.podAnnotations	A custom annotation to attach to all Scanner deployments.
customize.scanner.envVars	A custom environment variable for all Scanner containers.
customize.scanner-db.labels	A custom label to attach to all objects that Scanner DB creates.
customize.scanner-db.annotations	A custom annotation to attach to all objects that Scanner DB creates.
customize.scanner-db.podLabels	A custom label to attach to all Scanner DB deployments.

Parameter	Description
customize.scanner-db.podAnnotations	A custom annotation to attach to all Scanner DB deployments.
customize.scanner-db.envVars	A custom environment variable for all Scanner DB containers.

You can also use:

- the **customize.other.service/*.labels** and the **customize.other.service/*.annotations** parameters, to specify labels and annotations for all objects.
- or, provide a specific service name, for example, **customize.other.service/central-loadbalancer.labels** and **customize.other.service/central-loadbalancer.annotations** as parameters and set their value.

4.2.2.2.8. Advanced customization

IMPORTANT

The parameters specified in this section are for information only. Red Hat does not support Red Hat Advanced Cluster Security for Kubernetes instances with modified namespace and release names.

Parameter	Description
allowNonstandardNamespace	Use **true** to deploy Red Hat Advanced Cluster Security for Kubernetes into a namespace other than the default namespace **stackrox**.
allowNonstandardReleaseName	Use **true** to deploy Red Hat Advanced Cluster Security for Kubernetes with a release name other than the default **stackrox-central-services**.

4.2.3. Installing the central-services Helm chart

After you configure the **values-public.yaml** and **values-private.yaml** files, install the **central-services** Helm chart to deploy the centralized components (Central and Scanner).

Procedure

- Run the following command:

```
$ helm install -n stackrox --create-namespace \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> 1
```

1 Use the **-f** option to specify the paths for your YAML configuration files.

4.2.3.1. Changing configuration options after deploying the central-services Helm chart

You can make changes to any configuration options after you have deployed the **central-services** Helm chart.

Procedure

1. Update the **values-public.yaml** and **values-private.yaml** configuration files with new values.
2. Run the **helm upgrade** command and specify the configuration files using the **-f** option:

```
$ helm upgrade -n stackrox \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> \
  -f <path_to_values_private.yaml>
```

NOTE

You can also specify configuration values using the **--set** or **--set-file** parameters. However, these options are not saved, and it requires you to manually specify all the options again whenever you make changes.

4.2.4. Generating an init bundle

Before you install the **SecuredCluster** resource on a cluster, you must create an init bundle. The cluster that has **SecuredCluster** installed and configured then uses this bundle to authenticate with Central.

You can create an init bundle by using the **roxctl** CLI or from the RHACS portal.

4.2.4.1. Generating an init bundle by using the roxctl CLI

You can create an init bundle with secrets by using the **roxctl** CLI.

Prerequisites

You have configured the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables.

- Set the **ROX_API_TOKEN** and the **ROX_CENTRAL_ADDRESS** environment variables:

```
$ export ROX_API_TOKEN=<api_token>
```

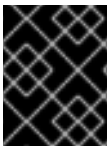
```
$ export ROX_CENTRAL_ADDRESS=<address>:<port_number>
```

Procedure

- Run the following command to generate a cluster init bundle containing secrets:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output cluster_init_bundle.yaml
```

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  central init-bundles generate <cluster_init_bundle_name> \
  --output-secrets cluster_init_bundle.yaml
```



IMPORTANT

Make sure that you store this bundle securely because it contains secrets. You can use the same bundle to set up multiple secured clusters.

Additional resources

- Installing the **roxctl** CLI

4.2.4.2. Generating an init bundle by using the RHACS portal

You can create an init bundle containing secrets by using the RHACS portal.

Procedure

1. Find the address of the RHACS portal based on your exposure method:
   a. For a route:

      ```
      $ oc get route central -n stackrox
      ```

   b. For a load balancer:

      ```
      $ oc get service central-loadbalancer -n stackrox
      ```

   c. For port forward:
      i. Run the following command:

         ```
         $ oc port-forward svc/central 18443:443 -n stackrox
         ```

      ii. Navigate to **https://localhost:18443/**.
2. On the RHACS portal, navigate to **Platform Configuration → Integrations**.
3. Navigate to the **Authentication Tokens** section and click on **Cluster Init Bundle**.
4. Click **Generate bundle**.
5. Enter a name for the cluster init bundle and click **Generate**.
6. Click **Download Helm Values File** to download the generated bundle.
7. Click **Download Kubernetes Secret File** to download the generated bundle.

IMPORTANT

Store this bundle securely because it contains secrets. You can use the same bundle to create multiple secured clusters.

Next Step

1. Use the OpenShift Container Platform CLI to create resources using the init bundle.
2. Install Red Hat Advanced Cluster Security for Kubernetes in all clusters that you want to monitor.

4.2.5. Configuring the secured-cluster-services Helm chart

This section describes Helm chart configuration parameters that you can use with the **helm install** and **helm upgrade** commands. You can specify these parameters by using the **--set** option or by creating YAML configuration files.

Create the following files for configuring the Helm chart for installing Red Hat Advanced Cluster Security for Kubernetes:

- Public configuration file **values-public.yaml**: Use this file to save all non-sensitive configuration options.
- Private configuration file **values-private.yaml**: Use this file to save all sensitive configuration options. Make sure that you store this file securely.

IMPORTANT

While using the **secured-cluster-services** Helm chart, do not modify the **values.yaml** file that is part of the chart.

4.2.5.1. Configuration parameters

Parameter	Description
clusterName	Name of your cluster.
centralEndpoint	Address of the Central endpoint, including the port number. If you are using a non-gRPC capable load balancer, use the WebSocket protocol by prefixing the endpoint address with **wss://**.
sensor.endpoint	Address of the Sensor endpoint including port number.
sensor.imagePullPolicy	Image pull policy for the Sensor container.
sensor.serviceTLS.cert	The internal service-to-service TLS certificate that Sensor uses.
sensor.serviceTLS.key	The internal service-to-service TLS certificate key that Sensor uses.
sensor.resources.requests.memory	The memory request for the Sensor container. Use this parameter to override the default value.

Parameter	Description
sensor.resources.requests.cpu	The CPU request for the Sensor container. Use this parameter to override the default value.
sensor.resources.limits.memory	The memory limit for the Sensor container. Use this parameter to override the default value.
sensor.resources.limits.cpu	The CPU limit for the Sensor container. Use this parameter to override the default value.
sensor.nodeSelector	Specify a node selector label as **label-key: label-value** to force Sensor to only schedule on nodes with the specified label.
sensor.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Sensor. This parameter is mainly used for infrastructure nodes.
image.main.name	The name of the **main** image.
image.collector.name	The name of the Collector image.
image.main.registry	Address of the registry you are using for the main image.
image.collector.registry	Address of the registry you are using for the Collector image.
image.main.pullPolicy	Image pull policy for **main** images.
image.collector.pullPolicy	Image pull policy for the Collector images.
image.main.tag	Tag of **main** image to use.
image.collector.tag	Tag of **collector** image to use.
collector.collectionMethod	Either **EBPF**, **KERNEL_MODULE**, or **NO_COLLECTION**.
collector.imagePullPolicy	Image pull policy for the Collector container.
collector.complianceImagePullPolicy	Image pull policy for the Compliance container.

Parameter	Description
collector.disableTaintTolerations	If you specify **false**, tolerations are applied to Collector, and the collector pods can schedule onto all nodes with taints. If you specify it as **true**, no tolerations are applied, and the collector pods are not scheduled onto nodes with taints.
collector.resources.requests.memory	The memory request for the Collector container. Use this parameter to override the default value.
collector.resources.requests.cpu	The CPU request for the Collector container. Use this parameter to override the default value.
collector.resources.limits.memory	The memory limit for the Collector container. Use this parameter to override the default value.
collector.resources.limits.cpu	The CPU limit for the Collector container. Use this parameter to override the default value.
collector.complianceResources.requests.memory	The memory request for the Compliance container. Use this parameter to override the default value.
collector.complianceResources.requests.cpu	The CPU request for the Compliance container. Use this parameter to override the default value.
collector.complianceResources.limits.memory	The memory limit for the Compliance container. Use this parameter to override the default value.
collector.complianceResources.limits.cpu	The CPU limit for the Compliance container. Use this parameter to override the default value.
collector.serviceTLS.cert	The internal service-to-service TLS certificate that Collector uses.
collector.serviceTLS.key	The internal service-to-service TLS certificate key that Collector uses.
admissionControl.listenOnCreates	This setting controls whether Kubernetes is configured to contact Red Hat Advanced Cluster Security for Kubernetes with **AdmissionReview** requests for workload creation events.

Parameter	Description
admissionControl.listenOnUpdates	When you set this parameter as **false**, Red Hat Advanced Cluster Security for Kubernetes creates the **ValidatingWebhookConfiguration** in a way that causes the Kubernetes API server not to send object update events. Since the volume of object updates is usually higher than the object creates, leaving this as **false** limits the load on the admission control service and decreases the chances of a malfunctioning admission control service.
admissionControl.listenOnEvents	This setting controls whether the cluster is configured to contact Red Hat Advanced Cluster Security for Kubernetes with **AdmissionReview** requests for Kubernetes **exec** and **portforward** events. Red Hat Advanced Cluster Security for Kubernetes does not support this feature on OpenShift Container Platform 3.11.
admissionControl.dynamic.enforceOnCreates	This setting controls whether Red Hat Advanced Cluster Security for Kubernetes evaluates policies; if it is disabled, all AdmissionReview requests are automatically accepted.
admissionControl.dynamic.enforceOnUpdates	This setting controls the behavior of the admission control service. You must specify **listenOnUpdates** as **true** for this to work.
admissionControl.dynamic.scanInline	If you set this option to **true**, the admission control service requests an image scan before making an admission decision. Since image scans take several seconds, enable this option only if you can ensure that all images used in your cluster are scanned before deployment (for example, by a CI integration during image build). This option corresponds to the **Contact image scanners** option in the RHACS Portal.
admissionControl.dynamic.disableBypass	Set it to **true** to disable bypassing the Admission Controller.
admissionControl.dynamic.timeout	The maximum time, in seconds, Red Hat Advanced Cluster Security for Kubernetes should wait while evaluating admission review requests. Use this to set request timeouts when you enable image scanning. If the image scan runs longer than the specified time, Red Hat Advanced Cluster Security for Kubernetes accepts the request.

Parameter	Description
admissionControl.resources.requests.memory	The memory request for the Admission Control container. Use this parameter to override the default value.
admissionControl.resources.requests.cpu	The CPU request for the Admission Control container. Use this parameter to override the default value.
admissionControl.resources.limits.memory	The memory limit for the Admission Control container. Use this parameter to override the default value.
admissionControl.resources.limits.cpu	The CPU limit for the Admission Control container. Use this parameter to override the default value.
admissionControl.nodeSelector	Specify a node selector label as **label-key: label-value** to force Admission Control to only schedule on nodes with the specified label.
admissionControl.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Admission Control. This parameter is mainly used for infrastructure nodes.
admissionControl.serviceTLS.cert	The internal service-to-service TLS certificate that Admission Control uses.
admissionControl.serviceTLS.key	The internal service-to-service TLS certificate key that Admission Control uses.
registryOverride	Use this parameter to override the default **docker.io** registry. Specify the name of your registry if you are using some other registry.
collector.disableTaintTolerations	If you specify **false**, tolerations are applied to Collector, and the Collector pods can schedule onto all nodes with taints. If you specify it as **true**, no tolerations are applied, and the Collector pods are not scheduled onto nodes with taints.
createUpgraderServiceAccount	Specify **true** to create the **sensor-upgrader** account. By default, Red Hat Advanced Cluster Security for Kubernetes creates a service account called **sensor-upgrader** in each secured cluster. This account is highly privileged but is only used during upgrades. If you do not create this account, you must complete future upgrades manually if the Sensor does not have enough permissions.

Parameter	Description
createSecrets	Specify **false** to skip the orchestrator secret creation for the Sensor, Collector, and Admission Controller.
collector.slimMode	Specify **true** if you want to use a slim Collector image for deploying Collector. Using slim Collector images requires Central to provide the matching kernel module or eBPF probe. If you are running Red Hat Advanced Cluster Security for Kubernetes in offline mode, you must download a kernel support package from stackrox.io and upload it to Central for slim Collectors to function. Otherwise, you must ensure that Central can access the online probe repository hosted at https://collector-modules.stackrox.io/.
sensor.resources	Resource specification for Sensor.
admissionControl.resources	Resource specification for Admission Controller.
collector.resources	Resource specification for Collector.
collector.complianceResources	Resource specification for Collector's Compliance container.
exposeMonitoring	If you set this option to **true**, Red Hat Advanced Cluster Security for Kubernetes exposes Prometheus metrics endpoints on port number 9090 for the Sensor, Collector, and the Admission Controller.
auditLogs.disableCollection	If you set this option to **true**, Red Hat Advanced Cluster Security for Kubernetes disables the audit log detection features used to detect access and modifications to configuration maps and secrets.
scanner.disable	If you set this option to **false**, Red Hat Advanced Cluster Security for Kubernetes deploys a lightweight scanner and Scanner DB in the secured cluster to allow scanning images on OpenShift Container Registry. Enabling Scanner is only supported on OpenShift. Defaults to **true**
scanner.dbTolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner DB.
scanner.replicas	Resource specification for Collector's Compliance container.

Parameter	Description
scanner.logLevel	Setting this parameter allows you to modify the scanner log level. Use this option only for troubleshooting purposes.
scanner.autoscaling.disable	If you set this option to **true**, Red Hat Advanced Cluster Security for Kubernetes disables autoscaling on the Scanner deployment.
scanner.autoscaling.minReplicas	The minimum number of replicas for autoscaling. Defaults to 2.
scanner.autoscaling.maxReplicas	The maximum number of replicas for autoscaling. Defaults to 5.
scanner.nodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner to only schedule on nodes with the specified label.
scanner.tolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner.
scanner.dbNodeSelector	Specify a node selector label as **label-key: label-value** to force Scanner DB to only schedule on nodes with the specified label.
scanner.dbTolerations	If the node selector selects tainted nodes, use this parameter to specify a taint toleration key, value, and effect for Scanner DB.
scanner.resources.requests.memory	The memory request for the Scanner container. Use this parameter to override the default value.
scanner.resources.requests.cpu	The CPU request for the Scanner container. Use this parameter to override the default value.
scanner.resources.limits.memory	The memory limit for the Scanner container. Use this parameter to override the default value.
scanner.resources.limits.cpu	The CPU limit for the Scanner container. Use this parameter to override the default value.
scanner.dbResources.requests.memory	The memory request for the Scanner DB container. Use this parameter to override the default value.
scanner.dbResources.requests.cpu	The CPU request for the Scanner DB container. Use this parameter to override the default value.

Parameter	Description
scanner.dbResources.limits.memory	The memory limit for the Scanner DB container. Use this parameter to override the default value.
scanner.dbResources.limits.cpu	The CPU limit for the Scanner DB container. Use this parameter to override the default value.

4.2.5.1.1. Environment variables

You can specify environment variables for Sensor and Admission Controller in the following format:

```
customize:
  envVars:
    ENV_VAR1: "value1"
    ENV_VAR2: "value2"
```

The **customize** setting allows you to specify custom Kubernetes metadata (labels and annotations) for all objects created by this Helm chart and additional pod labels, pod annotations, and container environment variables for workloads.

The configuration is hierarchical, in the sense that metadata defined at a more generic scope (for example, for all objects) can be overridden by metadata defined at a narrower scope (for example, only for the Sensor deployment).

4.2.6. Installing the secured-cluster-services Helm chart

After you configure the **values-public.yaml** and **values-private.yaml** files, install the **secured-cluster-services** Helm chart to deploy the per-cluster and per-node components (Sensor, Admission Controller, and Collector).

CAUTION

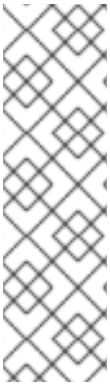
To install Collector on systems that have Unified Extensible Firmware Interface (UEFI) and that have Secure Boot enabled, you must use eBPF probes because kernel modules are unsigned, and the UEFI firmware cannot load unsigned packages. Collector identifies Secure Boot status at the start and switches to eBPF probes if required.

Procedure

- Run the following command:

```
$ helm install -n stackrox --create-namespace \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  -f <name_of_cluster_init_bundle.yaml> \
  -f <path_to_values_public.yaml> -f <path_to_values_private.yaml> (1)
```

(1) Use the **-f** option to specify the paths for your YAML configuration files.



NOTE

To deploy **secured-cluster-services** Helm chart by using a continuous integration (CI) system, pass the init bundle YAML file as an environment variable to the **helm install** command:

```
$ helm install ... -f <(echo "$INIT_BUNDLE_YAML_SECRET") 1
```

1 If you are using base64 encoded variables, use the **helm install … -f <(echo "$INIT_BUNDLE_YAML_SECRET" | base64 --decode)** command instead.

4.2.6.1. Changing configuration options after deploying the secured-cluster-services Helm chart

You can make changes to any configuration options after you have deployed the **secured-cluster-services** Helm chart.

Procedure

1. Update the **values-public.yaml** and **values-private.yaml** configuration files with new values.
2. Run the **helm upgrade** command and specify the configuration files using the **-f** option:

```
$ helm upgrade -n stackrox \
  stackrox-secured-cluster-services rhacs/secured-cluster-services \
  --reuse-values \ 1
  -f <path_to_values_public.yaml> \
  -f <path_to_values_private.yaml>
```

1 You must specify the **--reuse-values** parameter, otherwise the Helm upgrade command resets all previously configured settings.

NOTE

You can also specify configuration values using the **--set** or **--set-file** parameters. However, these options are not saved, and it requires you to manually specify all the options again whenever you make changes.

4.2.7. Verifying installation

After you complete the installation, run a few vulnerable applications and navigate to the RHACS portal to evaluate the results of security assessments and policy violations.

NOTE

The sample applications listed in the following section contain critical vulnerabilities and they are specifically designed to verify the build and deploy-time assessment features of Red Hat Advanced Cluster Security for Kubernetes.

To verify installation:

1. Find the address of the RHACS portal based on your exposure method:

a. For a route:

```
$ oc get route central -n stackrox
```

b. For a load balancer:

```
$ oc get service central-loadbalancer -n stackrox
```

c. For port forward:

i. Run the following command:

```
$ oc port-forward svc/central 18443:443 -n stackrox
```

ii. Navigate to **https://localhost:18443**/.

2. Using the OpenShift Container Platform CLI, create a new project:

```
$ oc new-project test
```

3. Start some applications with critical vulnerabilities:

```
$ oc run shell --labels=app=shellshock,team=test-team \
  --image=vulnerables/cve-2014-6271 -n test
$ oc run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes automatically scans these deployments for security risk and policy violations as soon as they are submitted to the cluster. Navigate to the RHACS portal to view the violations. You can log in to the RHACS portal by using the default username **admin** and the generated password.

CHAPTER 5. INSTALLING BY USING THE ROXCTL CLI

Red Hat Advanced Cluster Security for Kubernetes installs a set of services on your OpenShift Container Platform cluster. This topic describes the installation procedure for installing Red Hat Advanced Cluster Security for Kubernetes on your OpenShift Container Platform cluster by using the **roxctl** CLI.

WARNING

For production environments, Red Hat recommends Installing Red Hat Advanced Cluster Security for Kubernetes by using Helm charts. Do not use the **roxctl** install method unless you have a specific installation need that requires using this method.

High-level installation flow:

1. Install the **roxctl** CLI.
2. Use the **roxctl** CLI interactive installer to install the centralized components (Central and Scanner).
3. Install Sensor to monitor your cluster.

Before you install:

- Understand Red Hat Advanced Cluster Security for Kubernetes architecture .
- Review the prerequisites for installing Red Hat Advanced Cluster Security for Kubernetes .

5.1. INSTALLING THE ROXCTL CLI

To install Red Hat Advanced Cluster Security for Kubernetes you must install the **roxctl** CLI by downloading the binary. You can install **roxctl** on Linux, Windows, or macOS.

5.2. INSTALLING THE ROXCTL CLI ON LINUX

You can install the **roxctl** CLI binary on Linux by using the following procedure.

Procedure

1. Download the latest version of the **roxctl** CLI:

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.70.2/bin/Linux/roxctl
```

2. Make the **roxctl** binary executable:

```
$ chmod +x roxctl
```

3. Place the **roxctl** binary in a directory that is on your **PATH**:
   To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- Verify the **roxctl** version you have installed:

```
$ roxctl version
```

5.2.1. Installing the roxctl CLI on macOS

You can install the **roxctl** CLI binary on macOS by using the following procedure.

Procedure

1. Download the latest version of the **roxctl** CLI:

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.70.2/bin/Darwin/roxctl
```

2. Remove all extended attributes from the binary:

```
$ xattr -c roxctl
```

3. Make the **roxctl** binary executable:

```
$ chmod +x roxctl
```

4. Place the **roxctl** binary in a directory that is on your **PATH**:
   To check your **PATH**, execute the following command:

```
$ echo $PATH
```

Verification

- Verify the **roxctl** version you have installed:

```
$ roxctl version
```

5.2.2. Installing the roxctl CLI on Windows

You can install the **roxctl** CLI binary on Windows by using the following procedure.

Procedure

- Download the latest version of the **roxctl** CLI:

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.70.2/bin/Windows/roxctl.exe
```

Verification

- Verify the **roxctl** version you have installed:

```
$ roxctl version
```

5.3. INSTALLING CENTRAL

The main component of Red Hat Advanced Cluster Security for Kubernetes is called Central. You can install Central on OpenShift Container Platform by using the interactive installer. You deploy Central only once and you can monitor multiple separate clusters by using the same installation.

5.3.1. Using the interactive installer

Use the interactive installer to generate the required secrets, deployment configurations, and deployment scripts for your environment.

Procedure

1. Run the interactive install command:

```
$ roxctl central generate interactive
```

2. Press **Enter** to accept the default value for a prompt or enter custom values as required.

```
Enter path to the backup bundle from which to restore keys and certificates (optional):
Enter PEM cert bundle file (optional): 1
Enter administrator password (default: autogenerated):
Enter orchestrator (k8s, openshift): openshift
Enter the directory to output the deployment bundle to (default: "central-bundle"):
Enter the OpenShift major version (3 or 4) to deploy on (default: "0"): 4
Enter Istio version when deploying into an Istio-enabled cluster (leave empty when not
running Istio) (optional):
Enter the method of exposing Central (route, lb, np, none) (default: "none"): route 2
Enter main image to use (default: "stackrox.io/main:3.0.61.1"):
Enter whether to run StackRox in offline mode, which avoids reaching out to the Internet
(default: "false"):
Enter whether to enable telemetry (default: "true"):
Enter the deployment tool to use (kubectl, helm, helm-values) (default: "kubectl"):
Enter Scanner DB image to use (default: "stackrox.io/scanner-db:2.15.2"):
Enter Scanner image to use (default: "stackrox.io/scanner:2.15.2"):
Enter Central volume type (hostpath, pvc): pvc 3
Enter external volume name (default: "stackrox-db"):
Enter external volume size in Gi (default: "100"):
Enter storage class name (optional if you have a default StorageClass configured):
```

1. If you want to add a custom TLS certificate, provide the file path for the PEM-encoded certificate. When you specify a custom certificate the interactive installer also prompts you to provide a PEM private key for the custom certificate you are using.

2. To use the RHACS portal, you must expose Central by using a route, a load balancer or a node port.

3. If you plan to install Red Hat Advanced Cluster Security for Kubernetes on OpenShift Container Platform with a hostPath volume, you must modify the SELinux policy.

WARNING

On OpenShift Container Platform, for using a hostPath volume, you must modify the SELinux policy to allow access to the directory, which the host and the container share. It is because SELinux blocks directory sharing by default. To modify the SELinux policy, run the following command:

```
$ sudo chcon -Rt svirt_sandbox_file_t <full_volume_path>
```

However, Red Hat does not recommend modifying the SELinux policy, instead use PVC when installing on OpenShift Container Platform.

On completion, the installer creates a folder named central-bundle, which contains the necessary YAML manifests and scripts to deploy Central. In addition, it shows on-screen instructions for the scripts you need to run to deploy additional trusted certificate authorities, Central and Scanner, and the authentication instructions for logging into the RHACS portal along with the autogenerated password if you did not provide one when answering the prompts.

5.3.2. Running the Central installation scripts

After you run the interactive installer, you can run the **setup.sh** script to install Central.

Procedure

1. Run the **setup.sh** script to configure image registry access:

```
$ ./central-bundle/central/scripts/setup.sh
```

2. Create the necessary resources:

```
$ oc create -R -f central-bundle/central
```

3. Check the deployment progress:

```
$ oc get pod -n stackrox -w
```

4. After Central is running, find the RHACS portal IP address and open it in your browser. Depending on the exposure method you selected when answering the prompts, use one of the following methods to get the IP address.

Exposure method	Command	Address	Example
Route	**oc -n stackrox get route central**	The address under the **HOST/PORT** column in the output	**https://central-stackrox.example.route**

Exposure method	Command	Address	Example
Node Port	**oc get node -owide && oc -n stackrox get svc central-loadbalancer**	IP or hostname of any node, on the port shown for the service	**https://198.51.100.0:31489**
Load Balancer	**oc -n stackrox get svc central-loadbalancer**	EXTERNAL-IP or hostname shown for the service, on port 443	**https://192.0.2.0**
None	**central-bundle/central/scripts/port-forward.sh 8443**	**https://localhost:8443**	**https://localhost:8443**

NOTE

If you have selected autogenerated password during the interactive install, you can run the following command to see it for logging into Central:

```
$ cat central-bundle/password
```

5.4. INSTALLING SCANNER

You can configure Red Hat Advanced Cluster Security for Kubernetes to obtain image data from a variety of open-source and commercial image scanners.

However, Red Hat Advanced Cluster Security for Kubernetes also provides an image vulnerability scanner component, called Scanner. It enriches deployments with image vulnerability information.

Red Hat recommends deploying Scanner so that it can scan all images, including the images from public registries, for vulnerabilities. You can deploy the Scanner in the same cluster with Central.

Prerequisites

- You must configure your image registry to allow Scanner to download and scan images. Usually, image registry integrations are created automatically by Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. Run the following command to configure image registry access:

```
$ ./central-bundle/scanner/scripts/setup.sh
```

2. After the script finishes, run the following command to create the scanner service:

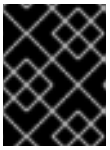
```
$ oc create -R -f central-bundle/scanner
```

5.5. INSTALLING SENSOR

To monitor a cluster, you must deploy Sensor. You must deploy Sensor into each cluster that you want to monitor. The following steps describe adding Sensor by using the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Clusters**.
2. Select **+ New Cluster**.
3. Specify a name for the cluster.
4. Provide appropriate values for the fields based on where you are deploying the Sensor.
   - If you are deploying Sensor in the same cluster, accept the default values for all the fields.
   - If you are deploying into a different cluster, replace **central.stackrox.svc:443** with a load balancer, node port, or other address, including the port number, that is accessible from the other cluster.
   - If you are using a non-gRPC capable load balancer, such as HAProxy, AWS Application Load Balancer (ALB), or AWS Elastic Load Balancing (ELB), use the WebSocket Secure (**wss**) protocol. To use **wss**:
     - Prefix the address with **wss://**.
     - Add the port number after the address, for example, **wss://stackrox-central.example.com:443**.
5. Click **Next** to continue with the Sensor setup.
6. Click **Download YAML File and Keys** to download the cluster bundle (zip archive).



IMPORTANT

The cluster bundle zip archive includes unique configurations and keys for each cluster. Do not reuse the same files in another cluster.

7. From a system that has access to the monitored cluster, unzip and run the **sensor** script from the cluster bundle:

```
$ unzip -d sensor sensor-<cluster_name>.zip
```

```
$ ./sensor/sensor.sh
```

If you get a warning that you do not have the required permissions to deploy Sensor, follow the on-screen instructions, or contact your cluster administrator for assistance.

After Sensor is deployed, it contacts Central and provides cluster information.

Verification

1. Return to the RHACS portal and check if the deployment is successful. If it is successful, a green checkmark appears under section #2. If you do not see a green checkmark, use the following command to check for problems:
   - On OpenShift Container Platform:

     ```
     $ oc get pod -n stackrox -w
     ```

   - On Kubernetes:

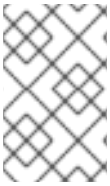
     ```
     $ kubectl get pod -n stackrox -w
     ```

2. Click **Finish** to close the window.

After installation, Sensor starts reporting security information to Red Hat Advanced Cluster Security for Kubernetes and the RHACS portal dashboard begins showing deployments, images, and policy violations from the cluster on which you have installed the Sensor.

5.6. VERIFYING INSTALLATION

After you complete the installation, run a few vulnerable applications and navigate to the RHACS portal to evaluate the results of security assessments and policy violations.



NOTE

The sample applications listed in the following section contain critical vulnerabilities and they are specifically designed to verify the build and deploy-time assessment features of Red Hat Advanced Cluster Security for Kubernetes.

To verify installation:

1. Find the address of the RHACS portal based on your exposure method:
   a. For a route:

      ```
      $ oc get route central -n stackrox
      ```

   b. For a load balancer:

      ```
      $ oc get service central-loadbalancer -n stackrox
      ```

   c. For port forward:
      i. Run the following command:

         ```
         $ oc port-forward svc/central 18443:443 -n stackrox
         ```

      ii. Navigate to **https://localhost:18443/**.
2. Using the OpenShift Container Platform CLI, create a new project:

   ```
   $ oc new-project test
   ```

3. Start some applications with critical vulnerabilities:

```
$ oc run shell --labels=app=shellshock,team=test-team \
  --image=vulnerables/cve-2014-6271 -n test
$ oc run samba --labels=app=rce \
  --image=vulnerables/cve-2017-7494 -n test
```

Red Hat Advanced Cluster Security for Kubernetes automatically scans these deployments for security risk and policy violations as soon as they are submitted to the cluster. Navigate to the RHACS portal to view the violations. You can log in to the RHACS portal by using the default username **admin** and the generated password.

5.7. ADDITIONAL RESOURCES

- Installing Red Hat Advanced Cluster Security for Kubernetes with customizations using Helm charts

CHAPTER 6. UNINSTALLING RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES

When you install Red Hat Advanced Cluster Security for Kubernetes, it creates:

- A namespace called **rhacs-operator** where the Operator is installed, if you chose the Operator method of installation
- A namespace called **stackrox**, or another namespace where you created the Central and SecuredCluster custom resources
- **PodSecurityPolicy** and Kubernetes role-based access control (RBAC) objects for all components
- Additional labels on namespaces, for use in generated network policies
- An application custom resource definition (CRD), if it does not exist

Uninstalling Red Hat Advanced Cluster Security for Kubernetes involves deleting all of these items.

6.1. DELETING NAMESPACE

You can delete the namespace that Red Hat Advanced Cluster Security for Kubernetes creates by using the OpenShift Container Platform or Kubernetes command-line interface.

Procedure

- Delete the **stackrox** namespace:
  - On OpenShift Container Platform:

```
$ oc delete namespace stackrox
```

  - On Kubernetes:

```
$ kubectl delete namespace stackrox
```

NOTE

If you installed RHACS in a different namespace, use the name of that namespace in the **delete** command.

6.2. DELETING GLOBAL RESOURCES

You can delete the global resources that Red Hat Advanced Cluster Security for Kubernetes creates, by using the OpenShift Container Platform or Kubernetes command-line interface.

Procedure

- Delete global resources:
  - On OpenShift Container Platform:

```
$ oc get clusterrole,clusterrolebinding,role,rolebinding,psp -o name | grep stackrox |
xargs oc delete --wait
```

```
$ oc delete scc -l "app.kubernetes.io/name=stackrox"
```

```
$ oc delete ValidatingWebhookConfiguration stackrox
```

- On Kubernetes:

```
$ kubectl get clusterrole,clusterrolebinding,role,rolebinding,psp -o name | grep stackrox |
xargs kubectl delete --wait
```

```
$ kubectl delete ValidatingWebhookConfiguration stackrox
```

6.3. DELETING LABELS AND ANNOTATIONS

You can delete the labels and annotations that Red Hat Advanced Cluster Security for Kubernetes creates, by using the OpenShift Container Platform or Kubernetes command-line interface.

Procedure

- Delete labels and annotations:
  - On OpenShift Container Platform:

```
$ for namespace in $(oc get ns | tail -n +2 | awk '{print $1}'); do      oc label namespace
$namespace namespace.metadata.stackrox.io/id-;      oc label namespace $namespace
namespace.metadata.stackrox.io/name-;      oc annotate namespace $namespace
modified-by.stackrox.io/namespace-label-patcher-;   done
```

  - On Kubernetes:

```
$ for namespace in $(kubectl get ns | tail -n +2 | awk '{print $1}'); do      kubectl label
namespace $namespace namespace.metadata.stackrox.io/id-;      kubectl label
namespace $namespace namespace.metadata.stackrox.io/name-;      kubectl annotate
namespace $namespace modified-by.stackrox.io/namespace-label-patcher-;   done
```