Red Hat Advanced Cluster Security for Kubernetes 3.73

Operating

Operating Red Hat Advanced Cluster Security for Kubernetes

Red Hat Advanced Cluster Security for Kubernetes 3.73 Operating

Operating Red Hat Advanced Cluster Security for Kubernetes

Legal Notice

Abstract

This document describes how to perform common operating tasks in Red Hat Advanced Cluster Security for Kubernetes, including using the dashboards, managing compliance, evaluating security risks, managing security policies and network policies, examining images for vulnerabilities, and responding to violations.

Table of Contents

CHAPTER 1. VIEWING THE DASHBOARD

The Red Hat Advanced Cluster Security for Kubernetes (RHACS) Dashboard provides quick access to the data you need. It contains additional navigation shortcuts and actionable widgets that are easy to filter and customize so that you can focus on the data that matters most to you. You can view information about levels of risk in your environment, compliance status, policy violations, and common vulnerabilities and exposures (CVEs) in images.

> **NOTE**
>
> When you open the RHACS portal for the first time, the Dashboard might be empty. After you deploy Sensor in at least one cluster, the Dashboard reflects the status of your environment.

The following sections describe the Dashboard components.

1.1. STATUS BAR

The **Status Bar** provides at-a-glance numerical counters for key resources. The counters reflect what is visible with your current access scope that is defined by the roles associated with your user profile. These counters are clickable, providing fast access to desired list view pages as follows:

Counter	Destination
Clusters	Platform Configuration → Clusters
Nodes	Configuration Management → Application & Infrastructure → Nodes
Violations	Violations main menu
Deployments	Configuration Management → Application & Infrastructure → Deployments
Images	Vulnerability Management → Dashboard → Images
Secrets	Configuration Management → Application & Infrastructure → Secrets

1.2. DASHBOARD FILTER

The Dashboard includes a top-level filter that applies simultaneously to all widgets. You can select one or more clusters, and one or more namespaces within selected clusters. When no clusters or namespaces are selected, the view automatically switches to **All**. Any change to the filter is immediately reflected by all widgets, limiting the data they present to the selected scope. The Dashboard filter does not affect the **Status Bar**.

1.3. WIDGET OPTIONS

Some widgets are customizable to help you focus on specific data. Widgets offer different controls that you can use to change how the data is sorted, filter the data, and customize the output of the widget.

Widgets offer two ways to customize different aspects:

- An **Options** menu, when present, provides specific options applicable to that widget.

- A **dynamic axis legend**, when present, provides a method to filter data by hiding one or more of the axis categories. For example, in the **Policy violations by category** widget, you can click on a severity to include or exclude violations of a selected severity from the data.

> **NOTE**
>
> Individual widget customization settings are short-lived and are reset to the system default upon leaving the Dashboard.

1.4. ACTIONABLE WIDGETS

The following sections describe the actionable widgets available in the Dashboard.

1.4.1. Policy violations by severity

This widget shows the distribution of violations across severity levels for the Dashboard-filtered scope. Clicking a **severity level** in the chart takes you to the **Violations** page, filtered for that severity and scope. It also lists the three most recent violations of a **Critical** level policy within the scope you defined in the Dashboard filter. Clicking a specific violation takes you directly to the **Violations** detail page for that violation.

1.4.2. Images at most risk

This widget lists the top six vulnerable images within the Dashboard-filtered scope, sorted by their computed risk priority, along with the number of critical and important CVEs they contain. Click on an image name to go directly to the **Image Findings** page under **Vulnerability Management**. Use the **Options** menu to focus on fixable CVEs, or further focus on active images.
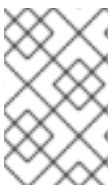
> **NOTE**
>
> When clusters or namespaces have been selected in the Dashboard filter, the data displayed is already filtered to active images, or images that are used by deployments within the filtered scope.

1.4.3. Deployments at most risk

This widget provides information about the top deployments at risk in your environment. It displays additional information such as the resource location (cluster and namespace) and the risk priority score. Additionally, you can click on a deployment to view risk information about the deployment; for example, its policy violations and vulnerabilities.

1.4.4. Aging images

Older images present a higher security risk because they can contain vulnerabilities that have already been addressed. If older images are active, they can expose deployments to exploits. You can use this widget to quickly assess your security posture and identify offending images. You can use the default ranges or customize the age intervals with your own values. You can view both inactive and active

images or use the Dashboard filter to focus on a particular area for active images. You can then click on an age group in this widget to view only those images in the **Vulnerability Management → Images** page.

1.4.5. Policy violations by category

This widget can help you gain insights about the challenges your organization is facing in complying with security policies, by analyzing which types of policies are violated more than others. The widget shows the five policy categories of highest interest. Explore the **Options** menu for different ways to slice the data. You can filter the data to focus exclusively on deploy or runtime violations.

You can also change the sorting mode. By default, the data is sorted by the number of violations within the highest severity first. Therefore, all categories with critical policies will appear before categories without critical policies. The other sorting mode considers the total number of violations regardless of severity. Because some categories contain no critical policies (for example, "Docker CIS"), the two sorting modes can provide significantly different views, offering additional insight.

Click on a severity level at the bottom of the graph to include or exclude that level from the data. Selecting different severity levels can result in a different top five selection or ranking order. Data is filtered to the scope selected by the Dashboard filter.

1.4.6. Compliance by standard

You can use the **Compliance by standard** widget with the Dashboard filter to focus on areas that matter to you the most. The widget lists the top or bottom six compliance benchmarks, depending on sort order. Select **Options** to sort by the coverage percentage. Click on one of the benchmark labels or graphs to go directly to the **Compliance Controls** page, filtered by the Dashboard scope and the selected benchmark.

> **NOTE**
>
> The **Compliance** widget shows details only after you run a compliance scan.

CHAPTER 2. MANAGING COMPLIANCE

By using Red Hat Advanced Cluster Security for Kubernetes you can assess, check, and report on the compliance status of your containerized infrastructure. You can run out-of-the-box compliance scans based on industry standards including:

- **CIS Benchmarks** (Center for Internet Security) for **Docker** and **Kubernetes**

- **HIPAA** (Health Insurance Portability and Accountability Act)

- **NIST Special Publication 800-190**and **800-53** (National Institute of Standards and Technology)

- **PCI DSS** (Payment Card Industry Data Security Standard)

- **OpenSCAP** (Open Security Content Automation Protocol): Available in RHACS for OpenShift Container Platform clusters when the Compliance Operator is installed and configured to provide results to RHACS

By scanning your environment based on these standards you can:

- Evaluate your infrastructure for regulatory compliance.

- Harden your Docker Engine and Kubernetes orchestrator.

- Understand and manage the overall security posture of your environment.

- Get a detailed view of compliance status for clusters, namespaces, and nodes.

2.1. VIEWING THE COMPLIANCE DASHBOARD

The compliance dashboard provides a high-level view of the compliance standards across all clusters, namespaces, and nodes in your environment.

The compliance dashboard includes charts and provides options to investigate a potential problem with compliance mandates. You can navigate to compliance scan results for a single cluster, namespace, or a node. Moreover, you can generate reports on the state of compliance within your containerized environment.

Procedure

- On the RHACS portal, select **Compliance** from the navigation menu.

> **NOTE**
>
> The first time you open the Compliance dashboard you will see a blank dashboard. You must run a compliance scan to populate the dashboard.

2.2. RUNNING A COMPLIANCE SCAN

Running a compliance scan checks the compliance status for your entire infrastructure across all compliance standards. When you run a compliance scan, Red Hat Advanced Cluster Security for Kubernetes takes a data snapshot of your environment. The data snapshot includes alerts, images, network policies, deployments, and related host-based data. Central collects the host-based data from

the Sensors running in your clusters. After that, Central collects more data from the compliance container running in each collector pod. The compliance container collects the following data about your environment:

- Configurations for Docker Daemon, Docker image, and Docker container.

- Information about Docker networks.

- Command-line arguments and processes for Docker, Kubernetes, and OpenShift Container Platform.

- Permissions of specific file paths.

- Configuration files for the core Kubernetes and OpenShift Container Platform services.

After the data collection is complete, Central performs checks on the data to determine results. You can view the results from the compliance dashboard and also generate compliance reports based on the results.

> **NOTE**
>
> In a compliance scan:
>
> - **Control** describes a single line item in an industry or regulatory compliance standard against which an auditor evaluates an information system for compliance with said standard. Red Hat Advanced Cluster Security for Kubernetes checks the evidence of compliance with a single control by completing one or more checks.
>
> - **Check** is the single test performed during a single control assessment.
>
> - Some controls have multiple checks associated with them. If any of the associated check fails for a control, the entire control state is marked as **Fail**.

Procedure

1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. Click **Scan environment**.

> **NOTE**
>
> Scanning the entire environment takes about 2 minutes to complete. This time might vary depending on the number of clusters and nodes in your environment.

2.3. VIEWING COMPLIANCE SCAN RESULTS

After you run a compliance scan, the compliance dashboard displays the results as the compliance status for your environment. You can view compliance violations directly from the dashboard, filter the details view, and drill down compliance standards to understand if your environment is compliant against specific benchmarks. This section explains how to view and filter compliance scan results.

You can use shortcuts to check the compliance status of clusters, namespaces, and nodes. Look for these shortcuts on the top of your compliance dashboard. By clicking these shortcuts you can view the

compliance snapshot and generate reports on the overall compliance of your clusters, namespaces, or nodes.

Compliance status

Status	Description
Fail	The compliance check failed.
Pass	The compliance check passed.
N/A	Red Hat Advanced Cluster Security for Kubernetes skipped the check because it was not applicable.
Info	The compliance check gathered data, but Red Hat Advanced Cluster Security for Kubernetes could not make a **Pass** or **Fail** determination.
Error	The compliance check failed due to a technical issue.

2.3.1. Viewing compliance status for clusters

You can view compliance status for all clusters or a single cluster from the compliance dashboard.

Procedure

- To view compliance status for all clusters in your environment:

    a. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

    b. Click **Clusters** on the compliance dashboard.

- To view compliance status for a specific cluster in your environment:

    a. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

    b. On the compliance dashboard, look for the **Passing standards by cluster** widget.

    c. In this widget, click on a cluster name to view its compliance status.

2.3.2. Viewing compliance status for namespaces

You can view compliance status for all namespaces or a single namespace from the compliance dashboard.

Procedure

- To view compliance status for all namespaces in your environment:

    1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. Click **Namespaces** on the compliance dashboard.

- To view compliance status for a specific namespace in your environment:

    1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

    2. Click **Namespaces** to open the namespaces details page.

    3. From the **Namespaces** table, click on a namespace. A side panel opens on the right.

    4. In the side panel, click on the name of the namespace to view its compliance status.

2.3.3. Viewing compliance status for a specific standard

Red Hat Advanced Cluster Security for Kubernetes supports NIST, PCI DSS, NIST, HIPAA, CIS for Kubernetes and CIS for Docker compliance standards. You can view all the compliance controls for a single compliance standard.

Procedure

1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. On the compliance dashboard, look for the **Passing standards across clusters cluster**widget.

3. In this widget, click on a standard to view information about all the controls associated with that standard.

> **NOTE**
>
> Many of the controls in CIS Docker refer to the configuration of the Docker engine on each Kubernetes node. Many CIS Docker controls are also best practices for building and using containers, and RHACS has policies to enforce their use. See "Managing security policies" in "Additional resources" for more information.

Additional resources

Managing security policies

2.3.4. Viewing compliance status for a specific control

You can view compliance status for a specific control for a selected standard.

Procedure

1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. On the compliance dashboard, look for the **Passing standards by cluster**widget.

3. In this widget, click on a standard to view information about all the controls associated with that standard.

4. From the **Controls** table, click on a control. A side panel opens on the right.

5. In the side panel, click on the name of the control to view its details.

2.4. FILTERING COMPLIANCE STATUS

Red Hat Advanced Cluster Security for Kubernetes search makes it easy to filter different combinations of data from the compliance dashboard. To focus your attention on a subset of clusters, industry standards, passing or failing controls, you can narrow the scope of the data visible on the compliance dashboard.

Procedure

1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. On the compliance dashboard, select either **Clusters**, or **Namespaces**, or **Nodes** to open the details page.

3. Enter your filtering criteria in the search bar and then press **Enter**.

2.5. GENERATING COMPLIANCE REPORTS

Red Hat Advanced Cluster Security for Kubernetes enables you to generate reports to keep track of the compliance status of your environment. You can use these reports to convey compliance status across various industry mandates to other stakeholders.

You can generate:

- **Executive reports** that focuses on the business aspect and includes charts and summary of compliance status in PDF format.

- **Evidence reports** that focuses on the technical aspect and includes detailed information in CSV format.

Procedure

1. Navigate to the RHACS portal and open the compliance dashboard by selecting **Compliance** from the navigation menu.

2. On the compliance dashboard, click **Export** on the top right side.

   - To generate an executive report, select **Download page as PDF**.

   - To generate an evidence report, select **Download Evidence as CSV**.

TIP

The **Export** option appears on all compliance pages and filtered views.

2.5.1. Evidence reports

You can export comprehensive compliance-related data from Red Hat Advanced Cluster Security for Kubernetes in CSV format as an evidence report. This evidence report contains detailed information about the compliance assessment, and it is tailored towards technical roles, such as compliance auditors, DevOps engineers, or security practitioners.

An evidence report contains the following information:

CSV field	Description
Standard	The compliance standard, for example, CIS Kubernetes.
Cluster	The name of the assessed cluster.
Namespace	The name of the namespace or project where the deployment exists.
Object Type	The Kubernetes entity type of the object. For example, **node**, **cluster**, **DaemonSet**, **Deployment**, or **StaticPod**.
Object Name	The name of the object which is a Kubernetes systems-generated string that uniquely identify objects. For example, **gke-setup-dev21380-default-pool-8e086a77-1jfq**.
Control	The control number as it appears in the compliance standard.
Control Description	Description about the compliance check that the control carries out.
State	Whether the compliance check passed or failed. For example, **Pass** or **Fail**.
Evidence	The explanation about why a specific compliance check failed or passed.
Assessment Time	The time and date when you ran the compliance scan.

2.6. SUPPORTED BENCHMARK VERSIONS

Red Hat Advanced Cluster Security for Kubernetes supports compliance checks against the following industry standards and regulatory frameworks:

Benchmark	Supported version
CIS Benchmarks (Center for Internet Security) for Docker and Kubernetes	CIS Kubernetes v1.5.0 and CIS Docker v1.2.0
HIPAA (Health Insurance Portability and Accountability Act)	HIPAA 164

Benchmark	Supported version
NIST (National Institute of Standards and Technology)	NIST Special Publication 800-190 and 800-53 Rev. 4
PCI DSS (Payment Card Industry Data Security Standard)	PCI DSS 3.2.1

CHAPTER 3. USING THE COMPLIANCE OPERATOR

3.1. USING THE COMPLIANCE OPERATOR WITH RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES

You can configure RHACS to use the Compliance Operator for compliance reporting and remediation with OpenShift Container Platform clusters. Results from the Compliance Operator can be reported in the RHACS Compliance Dashboard.

3.1.1. Installing the Compliance Operator

Install the Compliance Operator using Operator Hub.

Procedure

Install the Operator by performing the following steps:

1. Navigate in the web console to the **Operators → OperatorHub** page.

2. Enter **compliance operator** into the **Filter by keyword** box to find the Compliance Operator.

3. Select the **Compliance Operator** to view the details page.

4. Read the information about the Operator and click **Install**.

3.1.2. Configuring the ScanSettingBinding object

Create a **ScanSettingBinding** object in the **openshift-compliance** namespace to scan the cluster using the **cis** and **cis-node** profiles.

> **NOTE**
>
> This example uses **cis** and **cis-node** profiles, but OpenShift Container Platform provides additional profiles. See "Understanding the Compliance Operator" in the "Additional resources" section for more information.

Procedure

Select one of the following options:

- Use the CLI to create the YAML file and object. For example:

  a. Create a file called **sscan.yaml** using the following text:

  ```
  apiVersion: compliance.openshift.io/v1alpha1
  kind: ScanSettingBinding
  metadata:
    name: cis-compliance
  profiles:
    - name: ocp4-cis-node
      kind: Profile
      apiGroup: compliance.openshift.io/v1alpha1
    - name: ocp4-cis
      kind: Profile
  ```

```
        apiGroup: compliance.openshift.io/v1alpha1
    settingsRef:
     name: default
     kind: ScanSetting
     apiGroup: compliance.openshift.io/v1alpha1
```

b. Create the **ScanSettingBinding** object by running the following command:

```
$ oc create -f sscan.yaml -n openshift-compliance
```

If successful, the following message is displayed:

```
$ scansettingbinding.compliance.openshift.io/cis-compliance created
```

- Use the web console to create the object by performing the following steps:

  a. Change the active project to **openshift-compliance**.

  b. Click **+** to open the **Import YAML** window.

  c. Paste the YAML from the previous example and then click **Create**.

Additional resources

- [Understanding the Compliance Operator](#)

- [Compliance Operator scans](#) in OpenShift Container Platform

Optional: If you installed the Compliance Operator *after* installing RHACS, restart Sensor in the secured cluster by performing one of the following options:

- Run the following command:

```
$ oc -n stackrox delete pod -lapp=sensor
```

- In the OpenShift Container Platform web console, perform the following steps:

  a. Change the active project to **stackrox**.

  b. Navigate to **Workloads → Pods**.

  c. Locate the pod with the name starting with **sensor-**, and then click **Actions → Delete Pod.**

Verification

After performing these steps, run a compliance scan in RHACS and ensure that **ocp4-cis** and **ocp4-cis-node** results are displayed. See "Running a compliance scan" in the "Additional resources" section for more information.

Additional resources

- [Running a compliance scan in RHACS](#)

CHAPTER 4. EVALUATING SECURITY RISKS

Red Hat Advanced Cluster Security for Kubernetes assesses risk across your entire environment and ranks your running deployments according to their security risk. It also provides details about vulnerabilities, configurations, and runtime activities that require immediate attention.

4.1. RISK VIEW

The **Risk** view lists all deployments from all clusters, sorted by a multi-factor risk metric based on policy violations, image contents, deployment configuration, and other similar factors. Deployments at the top of the list present the most risk.

The **Risk** view shows list of deployments with following attributes for each row:

- **Name**: The name of the deployment.

- **Created**: The creation time of the deployment.

- **Cluster**: The name of the cluster where the deployment is running.

- **Namespace**: The namespace in which the deployment exists.

- **Priority**: A priority ranking based on severity and risk metrics.

In the **Risk** view, you can:

- Select a column heading to sort the violations in ascending or descending order.

- Use the filter bar to filter violations.

- Create a new policy based on the filtered criteria.

To view more details about the risks for a deployment, select a deployment in the **Risk** view.

4.1.1. Opening the risk view

You can analyze all risks in the **Risk** view and take corrective action.

Procedure

- Navigate to the RHACS portal and select **Risk** from the navigation menu.

4.2. CREATING A SECURITY POLICY FROM THE RISK VIEW

While evaluating risks in your deployments in the **Risk** view, when you apply local page filtering, you can create new security policies based on the filtering criteria you are using.

Procedure

1. Navigate to the RHACS portal and select **Risk** from the navigation menu.

2. Apply local page filtering criteria that you want to create a policy for.

3. Select **New Policy** and fill in the required fields to create a new policy.

4.2.1. Understanding how Red Hat Advanced Cluster Security for Kubernetes transforms the filtering criteria into policy criteria

When you create new security policies from the **Risk** view, based on the filtering criteria you use, not all criteria are directly applied to the new policy.

- Red Hat Advanced Cluster Security for Kubernetes converts the **Cluster**, **Namespace**, and **Deployment** filters to equivalent policy scopes.

  - Local page filtering on the **Risk** view combines the search terms by using the following methods:

    - Combines the search terms within the same category with an **OR** operator. For example, if the search query is **Cluster:A,B**, the filter matches deployments in **cluster A** or **cluster B**.

    - Combines the search terms from different categories with an **AND** operator. For example, if the search query is **Cluster:A+Namespace:Z**, the filter matches deployments in **cluster A** and in **namespace Z**.

  - When you add multiple scopes to a policy, the policy matches violations from any of the scopes.

    - For example, if you search for **(Cluster A OR Cluster B) AND (Namespace Z)** it results in two policy scopes, **(Cluster=A AND Namespace=Z)** OR **(Cluster=B AND Namespace=Z)**.

- Red Hat Advanced Cluster Security for Kubernetes drops or modifies filters that do not directly map to policy criteria and reports the dropped filters.

The following table lists how the filtering search attributes map to the policy criteria:

Search attribute	Policy criteria
Add Capabilities	Add Capabilities
Annotation	Disallowed Annotation
CPU Cores Limit	Container CPU Limit
CPU Cores Request	Container CPU Request
CVE	CVE
CVE Published On	✕ Dropped
CVE Snoozed	✕ Dropped
CVSS	CVSS
Cluster	↻ Converted to scope

Search attribute	Policy criteria
Component	Image Component (name)
Component Version	Image Component (version)
Deployment	↻ Converted to scope
Deployment Type	✕ Dropped
Dockerfile Instruction Keyword	Dockerfile Line (key)
Dockerfile Instruction Value	Dockerfile Line (value)
Drop Capabilities	✕ Dropped
Environment Key	Environment Variable (key)
Environment Value	Environment Variable (value)
Environment Variable Source	Environment Variable (source)
Exposed Node Port	✕ Dropped
Exposing Service	✕ Dropped
Exposing Service Port	✕ Dropped
Exposure Level	Port Exposure
External Hostname	✕ Dropped
External IP	✕ Dropped
Image	✕ Dropped
Image Command	✕ Dropped
Image Created Time	Days since image was created
Image Entrypoint	✕ Dropped
Image Label	Disallowed Image Label
Image OS	Image OS
Image Pull Secret	✕ Dropped

Search attribute	Policy criteria
Image Registry	Image Registry
Image Remote	Image Remote
Image Scan Time	Days since image was last scanned
Image Tag	Image Tag
Image Top CVSS	✕ Dropped
Image User	✕ Dropped
Image Volumes	✕ Dropped
Label	↻ Converted to scope
Max Exposure Level	✕ Dropped
Memory Limit (MB)	Container Memory Limit
Memory Request (MB)	Container Memory Request
Namespace	↻ Converted to scope
Namespace ID	✕ Dropped
Pod Label	✕ Dropped
Port	Port
Port Protocol	Protocol
Priority	✕ Dropped
Privileged	Privileged
Process Ancestor	Process Ancestor
Process Arguments	Process Arguments
Process Name	Process Name
Process Path	✕ Dropped
Process Tag	✕ Dropped

Search attribute	Policy criteria
Process UID	Process UID
Read Only Root Filesystem	Read-Only Root Filesystem
Secret	✕ Dropped
Secret Path	✕ Dropped
Service Account	✕ Dropped
Service Account Permission Level	Minimum RBAC Permission Level
Toleration Key	✕ Dropped
Toleration Value	✕ Dropped
Volume Destination	Volume Destination
Volume Name	Volume Name
Volume ReadOnly	Writable Volume
Volume Source	Volume Source
Volume Type	Volume Type

4.3. VIEWING RISK DETAILS

When you select a deployment in the **Risk** view, the **Risk Details** open in a panel on the right. The **Risk Details** panel shows detailed information grouped by multiple tabs.

4.3.1. Risk Indicators tab

The **Risk Indicators** tab of the **Risk Details** panel explains the discovered risks.

The **Risk Indicators** tab includes the following sections:

- **Policy Violations**: The names of the policies that are violated for the selected deployment.

- **Suspicious Process Executions**: Suspicious processes, arguments, and container names that the process ran in.

- **Image Vulnerabilities**: Images including total CVEs with their CVSS scores.

- **Service Configurations**: Aspects of the configurations that are often problematic, such as read-write (RW) capability, whether capabilities are dropped, and the presence of privileged containers.

- **Service Reachability**: Container ports exposed inside or outside the cluster.

- **Components Useful for Attackers** Discovered software tools that are often used by attackers.

- **Number of Components in Image**: The number of packages found in each image.

- **Image Freshness**: Image names and age, for example, **285 days old**.

- **RBAC Configuration**: The level of permissions granted to the deployment in Kubernetes role-based access control (RBAC).

> **NOTE**
>
> Not all sections are visible in the **Risk Indicators** tab. Red Hat Advanced Cluster Security for Kubernetes displays only relevant sections that affect the selected deployment.

4.4. DEPLOYMENT DETAILS TAB

The sections in the **Deployment Details** tab of the **Deployment Risk** panel provide more information so you can make appropriate decisions on how to address the discovered risk.

4.4.1. Overview section

The **Overview** section shows details about the following:

- **Deployment ID**: An alphanumeric identifier for the deployment.

- **Namespace**: The Kubernetes or OpenShift Container Platform namespace in which the deployment exists.

- **Updated**: A timestamp with date for when the deployment was updated.

- **Deployment Type**: The type of deployment, for example, **Deployment** or **DaemonSet**.

- **Replicas**: The number of pods deployed for this deployment.

- **Labels**: The key-value labels attached to the Kubernetes or OpenShift Container Platform application.

- **Cluster**: The name of the cluster where the deployment is running.

- **Annotations**: The Kubernetes annotations for the deployment.

- **Service Account**: Represents an identity for processes that run in a pod. When a process is authenticated through a service account, it can contact the Kubernetes API server and access cluster resources. If a pod does not have an assigned service account, it gets the default service account.

4.4.2. Container configuration section

The container configuration section shows details about the following:

- **Image Name**: The name of the image that is deployed.

- **Resources**

- - CPU Request (cores): The number of CPUs requested by the container.

  - CPU Limit (cores): The maximum number of CPUs the container can use.

  - Memory Request (MB): The memory size requested by the container.

  - Memory Limit (MB): The maximum amount of memory the container can use without being killed.

- Mounts

  - Name: The name of the mount.

  - Source: The path from where the data for the mount comes.

  - Destination: The path to which the data for the mount goes.

  - Type: The type of the mount.

- Secrets: The names of Kubernetes secrets used in the deployment, and basic details for secret values that are X.509 certificates.

4.4.3. Security context section

The **Security Context** section shows details about the following:

- Privileged: Lists **true** if the container is privileged.

4.5. PROCESS DISCOVERY TAB

The **Process Discovery** tab provides a comprehensive list of all binaries that have been executed in each container in your environment, summarized by deployment.

The process discovery tab shows details about the following:

- Binary Name: The name of the binary that was executed.

- Container: The container in the deployment in which the process executed.

- Arguments: The specific arguments that were passed with the binary.

- Time: The date and time of the most recent time the binary was executed in a given container.

- Pod ID: The identifier of the pod in which the container resides.

- UID: The Linux user identity under which the process executed.

Use the **Process Name:<name>** query in the filter bar to find specific processes.

4.5.1. Event timeline section

The **Event Timeline** section in the **Process Discovery** tab provides an overview of events for the selected deployment. It shows the number of policy violations, process activities, and container termination or restart events.

You can select **Event Timeline** to view more details.

The **Event Timeline** modal box shows events for all pods for the selected deployment.

The events on the timeline are categorized as:

- Process activities

- Policy violations

- Container restarts

- Container terminations

The events appear as icons on a timeline. To see more details about an event, hold your mouse pointer over the event icon. The details appear in a tooltip.

- Click **Show Legend** to see which icon corresponds to which type of event.

- Select **Export → Download PDF** or **Export → Download CSV** to download the event timeline information.

- Select the **Show All** drop-down menu to filter which type of events are visible on the timeline.

- Click on the expand icon to see events separately for each container in the selected pod.

All events in the timeline are also visible in the minimap control at the bottom. The minimap controls the number of events visible in the event timeline. You can change the events shown in the timeline by modifying the highlighted area on the minimap. To do this, decrease the highlighted area from left or right sides (or both), and then drag the highlighted area.

> **NOTE**
>
> - When containers restart, Red Hat Advanced Cluster Security for Kubernetes:
>
>   - Shows information about container termination and restart events for up to 10 inactive container instances for each container in a pod. For example, for a pod with two containers **app** and **sidecar**, Red Hat Advanced Cluster Security for Kubernetes keeps activity for up to 10 **app** instances and up to 10 **sidecar** instances.
>
>   - Does not track process activities associated with the previous instances of the container.
>
> - Red Hat Advanced Cluster Security for Kubernetes only shows the most recent execution of each (process name, process arguments, UID) tuple for each pod.
>
> - Red Hat Advanced Cluster Security for Kubernetes shows events only for the active pods.
>
> - Red Hat Advanced Cluster Security for Kubernetes adjusts the reported timestamps based on time reported by Kubernetes and the Collector. Kubernetes timestamps use second-based precision, and it rounds off the time to the nearest second. However, the Collector uses more precise timestamps. For example, if Kubernetes reports the container start time as **10:54:48**, and the Collector reports a process in that container started at **10:54:47.5349823**, Red Hat Advanced Cluster Security for Kubernetes adjusts the container start time to **10:54:47.5349823**.

4.6. USING PROCESS BASELINES

You can minimize risk by using process baselining for infrastructure security. With this approach, Red Hat Advanced Cluster Security for Kubernetes first discovers existing processes and creates a baseline. Then it operates in the default deny-all mode and only allows processes listed in the baseline to run.

Process baselines
When you install Red Hat Advanced Cluster Security for Kubernetes, there is no default process baseline. As Red Hat Advanced Cluster Security for Kubernetes discovers deployments, it creates a process baseline for every container type in a deployment. Then it adds all discovered processes to their own process baselines.

Process baseline states
During the process discovery phase, all baselines are in an unlocked state.

In an **unlocked** state:

- When Red Hat Advanced Cluster Security for Kubernetes discovers a new process, it adds that process to the process baseline.

- Processes do not show up as risks and do not trigger any violations.

After an hour from when Red Hat Advanced Cluster Security for Kubernetes receives the first process indicator from a container in a deployment, it finishes the process discovery phase. At this point:

- Red Hat Advanced Cluster Security for Kubernetes stops adding processes to the process baselines.

- New processes that are not in the process baseline show up as risks, but they do not trigger any violations.

To generate violations, you must manually lock the process baseline.

In a **locked** state:

- Red Hat Advanced Cluster Security for Kubernetes stops adding processes to the process baselines.

- New processes that are not in the process baseline trigger violations.

Independent of the locked or unlocked baseline state, you can always add or remove processes from the baseline.
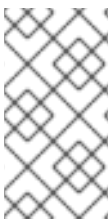
> **NOTE**
>
> For a deployment, if each pod has multiple containers in it, Red Hat Advanced Cluster Security for Kubernetes creates a process baseline for each container type. For such a deployment, if some baselines are locked and some are unlocked, the baseline status for that deployment shows up as **Mixed**.

4.6.1. Viewing the process baselines

You can view process baselines from the **Risk** view.

Procedure

1. In the RHACS portal, select **Risk** from the navigation menu.

2. Select a deployment from the list of deployments in the default **Risk** view. Deployment details open in a panel on the right.

3. In the **Deployment** details panel, select the **Process Discovery** tab.

4. The process baselines are visible under the **Spec Container Baselines** section.

4.6.2. Adding a process to the baseline

You can add a process to the baseline.

Procedure

1. In the RHACS portal, select **Risk** from the navigation menu.

2. Select a deployment from the list of deployments in the default **Risk** view. Deployment details open in a panel on the right.

3. In the **Deployment** details panel, select the **Process Discovery** tab.

4. Under the **Running Processes** section, click the **Add** icon for the process you want to add to the process baseline.
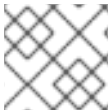
> **NOTE**
>
> The **Add** icon is available only for the processes that are not in the process baseline.

4.6.3. Removing a process from the baseline

You can remove a process from the baseline.

Procedure

1. In the RHACS portal, select **Risk** from the navigation menu.

2. Select a deployment from the list of deployments in the default **Risk** view. Deployment details open in a panel on the right.

3. In the **Deployment** details panel, select the **Process Discovery** tab.

4. Under the **Spec Container baselines** section, click the **Remove** icon for the process you want to remove from the process baseline.

4.6.4. Locking and unlocking the process baselines

You can **Lock** the baseline to trigger violations for all processes not listed in the baseline and **Unlock** the baseline to stop triggering violations.

Procedure

1. In the RHACS portal, select **Risk** from the navigation menu.

2. Select a deployment from the list of deployments in the default **Risk** view. Deployment details open in a panel on the right.

3. In the **Deployment** details panel, select the **Process Discovery** tab.

4. Under the **Spec Container baselines** section:

   - Click the **Lock** icon to trigger violations for processes that are not in the baseline.

   - Click the **Unlock** icon to stop triggering violations for processes that are not in the baseline.

CHAPTER 5. USING ADMISSION CONTROLLER ENFORCEMENT

Red Hat Advanced Cluster Security for Kubernetes works with Kubernetes admission controllers and OpenShift Container Platform admission plugins to allow you to enforce security policies before Kubernetes or OpenShift Container Platform creates workloads, for example, deployments, daemon sets or jobs.

The Red Hat Advanced Cluster Security for Kubernetes admission controller prevents users from creating workloads that violate policies you configure in Red Hat Advanced Cluster Security for Kubernetes. Beginning from the Red Hat Advanced Cluster Security for Kubernetes version 3.0.41, you can also configure the admission controller to prevent updates to workloads that violate policies.

Red Hat Advanced Cluster Security for Kubernetes uses the **ValidatingAdmissionWebhook** controller to verify that the resource being provisioned complies with the specified security policies. To handle this, Red Hat Advanced Cluster Security for Kubernetes creates a **ValidatingWebhookConfiguration** which contains multiple webhook rules.

When the Kubernetes or OpenShift Container Platform API server receives a request that matches one of the webhook rules, the API server sends an **AdmissionReview** request to Red Hat Advanced Cluster Security for Kubernetes. Red Hat Advanced Cluster Security for Kubernetes then accepts or rejects the request based on the configured security policies.

> **NOTE**
>
> To use admission controller enforcement on OpenShift Container Platform, you need the Red Hat Advanced Cluster Security for Kubernetes version 3.0.49 or newer.

5.1. UNDERSTANDING ADMISSION CONTROLLER ENFORCEMENT

If you intend to use admission controller enforcement, consider the following:

- **API latency**: Using admission controller enforcement increases Kubernetes or OpenShift Container Platform API latency because it involves additional API validation requests. Many standard Kubernetes libraries, such as fabric8, have short Kubernetes or OpenShift Container Platform API timeouts by default. Also, consider API timeouts in any custom automation you might be using.

- **Image scanning**: You can choose whether the admission controller scans images while reviewing requests by setting the **Contact Image Scanners** option in the cluster configuration panel.

  - If you enable this setting, Red Hat Advanced Cluster Security for Kubernetes contacts the image scanners if the scan or image signature verification results are not already available, which adds considerable latency.

  - If you disable this setting, the enforcement decision only considers image scan criteria if cached scan and signature verfication results are available.

- You can use admission controller enforcement for:

  - Options in the pod **securityContext**.

  - Deployment configurations.

  - Image components and vulnerabilities.

- You cannot use admission controller enforcement for:

  - Any runtime behavior, such as processes.

  - Any policies based on port exposure.

- The admission controller might fail if there are connectivity issues between the Kubernetes or OpenShift Container Platform API server and Red Hat Advanced Cluster Security for Kubernetes Sensor. To resolve this issue, delete the **ValidatingWebhookConfiguration** object as described in the disabling admission controller enforcement section.

- If you have deploy-time enforcement enabled for a policy and you enable the admission controller, Red Hat Advanced Cluster Security for Kubernetes attempts to block deployments that violate the policy. If a non-compliant deployment is not rejected by the admission controller, for example, in case of a timeout, Red Hat Advanced Cluster Security for Kubernetes still applies other deploy-time enforcement mechanisms, such as scaling to zero replicas.

5.2. ENABLING ADMISSION CONTROLLER ENFORCEMENT

You can enable admission controller enforcement from the **Clusters** view when you install Sensor or edit an existing cluster configuration.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Clusters**.

2. Select an existing cluster from the list or select **+ New Cluster**.

3. In the cluster configuration panel, enter the details for your cluster.

4. Red Hat recommends that you only turn on the **Configure Admission Controller Webhook to listen on creates** toggle if you are planning to use the admission controller to enforce on object create events.

5. Red Hat recommends that you only turn on the **Configure Admission Controller Webhook to listen on updates** toggle if you are planning to use the admission controller to enforce on update events.

6. Red Hat recommends that you only turn on the **Enable Admission Controller Webhook to listen on exec and port-forward events** toggle if you are planning to use the admission controller to enforce on pod execution and pod port forwards events.

7. Configure the following options:

   - **Enforce on Object Creates**: This toggle controls the behavior of the admission control service. You must have the **Configure Admission Controller Webhook to listen on creates** toggle turned on for this to work.

   - **Enforce on Object Updates**: This toggle controls the behavior of the admission control service. You must have the **Configure Admission Controller Webhook to listen on updates** toggle turned on for this to work.

8. Select **Next**.

9. In the **Download files** section, select **Download YAML Files and Keys**

NOTE

When enabling admission controller for an existing cluster, if you make any changes in the:

- **Static Configuration** section, you must download the YAML files and redeploy the Sensor.

- **Dynamic Configuration** section, you can skip downloading the files and deployment, as Red Hat Advanced Cluster Security for Kubernetes automatically syncs the Sensor and applies the changes.

10. Select **Finish**.

Verification

- After you provision a new cluster with the generated YAML, run the following command to verify if admission controller enforcement is configured correctly:

  ```
  $ oc get ValidatingWebhookConfiguration ❶
  ```

  ❶   If you use Kubernetes, enter **kubectl** instead of **oc**.

  Example output

  ```
  NAME        CREATED AT
  stackrox    2019-09-24T06:07:34Z
  ```

5.3. BYPASSING ADMISSION CONTROLLER ENFORCEMENT

To bypass the admission controller, add the **admission.stackrox.io/break-glass** annotation to your configuration YAML. Bypassing the admission controller triggers a policy violation which includes deployment details. Red Hat recommends providing an issue-tracker link or some other reference as the value of this annotation so that others can understand why you bypassed the admission controller.

5.4. DISABLING ADMISSION CONTROLLER ENFORCEMENT

You can disable admission controller enforcement from the **Clusters** view on the Red Hat Advanced Cluster Security for Kubernetes (RHACS) portal.

Procedure

1. On the RHACS portal, select **Platform Configuration → Clusters**.

2. Select an existing cluster from the list.

3. Turn off the **Enforce on Object Creates** and **Enforce on Object Updates** toggles in the **Dynamic Configuration** section.

4. Select **Next**.

5. Select **Finish**.

5.4.1. Disabling associated policies

You can turn off the enforcement on relevant policies, which in turn instructs the admission controller to skip enforcements.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. Disable enforcement on the default policies:

   - In the policies view, scroll down and select the power icon next to the **Kubernetes Actions: Exec into Pod** policy to disable that policy.

   - In the policies view, scroll down and select the power icon next to the **Kubernetes Actions: Port Forward to Pod** policy to disable that policy.

3. Disable enforcement on any other custom policies that you have created by using criteria from the default **Kubernetes Actions: Port Forward to Pod** and **Kubernetes Actions: Exec into Pod** policies.

5.4.2. Disabling the webhook

You can disable admission controller enforcement from the **Clusters** view on the RHACS portal.
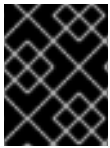
> **IMPORTANT**
>
> If you disable the admission controller by turning off the webhook, you must redeploy the Sensor bundle.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Clusters**.

2. Select an existing cluster from the list.

3. Turn off the **Enable Admission Controller Webhook to listen on exec and port-forward events** toggle in the **Static Configuration** section.

4. Select **Next** to continue with Sensor setup.

5. Click **Download YAML File and Keys**.

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

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

   ```
   $ ./sensor/sensor.sh
   ```
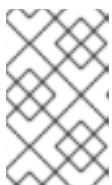
> **NOTE**
>
> If you get a warning that you do not have the required permissions to deploy the sensor, follow the on-screen instructions, or contact your cluster administrator for assistance.

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

7. 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
```

8. Select **Finish**.
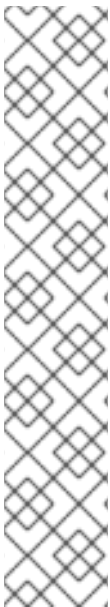
> **NOTE**
>
> When you disable the admission controller, Red Hat Advanced Cluster Security for Kubernetes does not delete the **ValidatingWebhookConfiguration**. However, instead of checking requests for violations, it accepts all **AdmissionReview** requests.
>
> To remove the **ValidatingWebhookConfiguration** object, run the following command in the secured cluster:
>
> - On OpenShift Container Platform:
>
> ```
> $ oc delete ValidatingWebhookConfiguration/stackrox
> ```
>
> - On Kubernetes:
>
> ```
> $ kubectl delete ValidatingWebhookConfiguration/stackrox
> ```

5.5. VALIDATINGWEBHOOKCONFIGURATION YAML FILE CHANGES

With Red Hat Advanced Cluster Security for Kubernetes you can enfore security policies on:

- Object creation

- Object update

- Pod execution

- Pod port forward

If Central or Sensor is unavailable

The admission controller requires an initial configuration from Sensor to work. Kubernetes or OpenShift Container Platform saves this configuration, and it remains accessible even if all admission control service replicas are rescheduled onto other nodes. If this initial configuration exists, the admission controller enforces all configured deploy–time policies.

If Sensor or Central becomes unavailable later:

- you will not be able to run image scans, or query information about cached image scans. However, admission controller enforcement still functions based on the available information gathered before the timeout expires, even if the gathered information is incomplete.

- you will not be able to disable the admission controller from the RHACS portal or modify enforcement for an existing policy as the changes will not get propagated to the admission control service.

> **NOTE**
>
> If you need to disable admission control enforcement, you can delete the validating webhook configuration by running the following command:
>
> - On OpenShift Container Platform:
>
>   ```
>   $ oc delete ValidatingWebhookConfiguration/stackrox
>   ```
>
> - On Kubernetes:
>
>   ```
>   $ kubectl delete ValidatingWebhookConfiguration/stackrox
>   ```

Make the admission controller more reliable

Red Hat recommends that you schedule the admission control service on the control plane and not on worker nodes. The deployment YAML file includes a soft preference for running on the control plane, however it is not enforced.

By default, the admission control service runs 3 replicas. To increase reliability, you can increase the replicas by running the following command:

```
$ oc -n stackrox scale deploy/admission-control --replicas=<number_of_replicas>
```
1

1 If you use Kubernetes, enter **kubectl** instead of **oc**.

Using with the roxctl CLI

You can use the following options when you generate a Sensor deployment YAML file:

- **--admission-controller-listen-on-updates**: If you use this option, Red Hat Advanced Cluster Security for Kubernetes generates a Sensor bundle with a **ValidatingWebhookConfiguration** pre-configured to receive update events from the Kubernetes or OpenShift Container Platform API server.

- **--admission-controller-enforce-on-updates**: If you use this option, Red Hat Advanced Cluster Security for Kubernetes configures Central such that the admission controller also enforces security policies object updates.

Both these options are optional, and are **false** by default.

CHAPTER 6. MANAGING SECURITY POLICIES

Red Hat Advanced Cluster Security for Kubernetes allows you to use out-of-the-box security policies and define custom multi-factor policies for your container environment. Configuring these policies enables you to automatically prevent high-risk service deployments in your environment and respond to runtime security incidents.

6.1. USING DEFAULT SECURITY POLICIES

Red Hat Advanced Cluster Security for Kubernetes includes a set of default policies that provide broad coverage to identify security issues and ensure best practices for security in your environment.

To view the default policies:

- On the RHACS portal, navigate to **Platform Configuration → Policies**.

The **Policies** view lists the default policies and includes the following parameters for each policy:

- **Policy**: A name for the policy.

- **Description**: A longer, more detailed description of the alert for the policy.

- **Status**: The current status of the policy, either **Enabled** or **Disabled**.

- **Notifiers**: The list of notifiers that are configured for the policy.

- **Severity**: A ranking of the policy, either critical, high, medium, or low, for the amount of attention required.

- **Lifecycle**: The phase of the container lifecycle (build, deploy, or runtime) that this policy applies to, and the phase at which enforcement applies, when the policy is enabled.

The default policies have preconfigured parameters and belong to categories such as:

- Anomalous Activity

- Cryptocurrency Mining

- DevOps Best Practices

- Kubernetes

- Network Tools

- Package Management

- Privileges

- Security Best Practices

- System Modification

- Vulnerability Management

You can edit these categories and create your own categories. When you create your own category, a new widget displays information about that category on the dashboard.
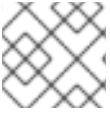
NOTE

You cannot delete default policies or edit policy criteria for default policies.

6.2. MODIFYING EXISTING SECURITY POLICIES

You can edit the policies you have created and the existing default policies provided by Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. From the **Policies** page, select the policy you want to edit.

3. Select **Actions → Edit policy**.

4. Modify the **Policy details**. You can modify the policy name, severity, categories, description, rationale, and guidance. You can also attach notifiers to the policy by selecting from the available **Notifiers** under the **Attach notifiers** section.

5. Click **Next**.

6. In the **Policy behavior** section, select the **Lifecycle stages** and **Event sources** for the policy.

7. Select a **Response method** to address violations for the policy.

8. Click **Next**.

9. In the **Policy criteria** section, expand the categories under the **Drag out policy fields** section. Use the drag-and-drop policy fields to specify logical conditions for the policy criteria.

   NOTE

   You cannot edit policy criteria for default policies.

10. Click **Next**.

11. In the **Policy scope** section, modify **Restrict by scope**, **Exclude by scope**, and **Exclude images** settings.

12. Click **Next**.

13. In the **Review policy** section, preview the policy violations.

14. Click **Save**.

Additional resources

- [Creating a security policy from the system policies view](#)

6.3. CREATING POLICY CATEGORIES

You can create new policy categories from the system policies view.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. From the **Policies** page, select the policy you want to edit.

3. Select **Actions → Edit policy**.

4. Enter a new category name in the **Categories** field and then click **Create <category>**.

5. Click the **Review policy** section heading.

6. Click **Save**.

After you save the configuration, the new category displays on the dashboard if there are any violations for policies in the category.

Additional resources

- [Creating a security policy from the system policies view](#)

6.4. CREATING CUSTOM POLICIES

In addition to using the default policies, you can also create custom policies in Red Hat Advanced Cluster Security for Kubernetes.

To build a new policy, you can clone an existing policy or create a new one from scratch.

- You can also create policies based on the filter criteria in the **Risk** view in the RHACS portal.

- You can also use **AND**, **OR**, and **NOT** logical operators for policy criteria to create advanced policies.

6.4.1. Creating a security policy from the system policies view

You can create new security policies from the system policies view.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. Click **Create policy**.

3. Enter the following details about your policy in the **Policy details** section:

   - Enter a **Name** for the policy.

   - Optional: Attach notifiers to the policy by selecting from the available **Notifiers** under the **Attach notifiers** section.

   > **NOTE**
   >
   > You must integrate Red Hat Advanced Cluster Security for Kubernetes with your notification provider, for example, webhooks, Jira, PagerDuty, Splunk, or others before you can forward alerts.

- Select a **Severity** level for this policy, either **Critical**, **High**, **Medium**, or **Low**.

- Select policy **Categories** you want to apply to this policy.

- Enter details about the policy in the **Description** box.

- Enter an explanation about why the policy exists in the **Rationale** box.

- Enter steps to resolve violations of this policy in the **Guidance** box.

- Optional: Under the **MITRE ATT&CK** section, select the tactics and the techniques you want to specify for the policy.

  a. Click **Add tactic**, and then select a tactic from the dropdown list.

  b. Click the **Add technique** to add techniques for the selected tactic. You can specify multiple techniques for a tactic.

4. Click **Next**.

5. In the **Policy behavior** section, select the **Lifecycle stages** and **Event sources (Runtime lifecycle only)** for the policy.

   - Choose **Lifecycle Stages** to which your policy is applicable, from **Build**, **Deploy**, or **Runtime**. You can select more than one stage.

     - Build-time policies apply to image fields such as CVEs and Dockerfile instructions.

     - Deploy-time policies can include all build-time policy criteria but they can also include data from your cluster configurations, such as running in privileged mode or mounting the Docker socket.

     - Runtime policies can include all build-time and deploy-time policy criteria but they can also include data about process executions during runtime.

6. For **Response method**, either select:

   a. **Inform** to include the violation in the violations list.

   b. Or select **Inform and enforce** to enforce actions.

     - Choose the enforcement behavior for the policy. It is only available for the stages you select when configuring **Lifecycle Stages**. Select **ON** (enable) to enforce policy and report a violation, and **OFF** (disable) to only report a violation. The enforcement behavior is different for each lifecycle stage.

       - **Build** - Red Hat Advanced Cluster Security for Kubernetes fails your continuous integration (CI) builds when images match the conditions of the policy.

       - **Deploy** - Red Hat Advanced Cluster Security for Kubernetes blocks creation of deployments that match the conditions of the policy. In clusters with admission controller enforcement, the Kubernetes or OpenShift Container Platform API server blocks all noncompliant deployments. In other clusters, Red Hat Advanced Cluster Security for Kubernetes edits noncompliant deployments to prevent pods from being scheduled.

       - **Runtime** - Red Hat Advanced Cluster Security for Kubernetes kills all pods that match the conditions of the policy or blocks the action taken on the pods.

> **WARNING**
>
> Policy enforcement can impact running applications or development processes. Before you enable enforcement options, inform all stakeholders and plan about how to respond to automated enforcement actions.

7. Click **Next**.

8. In the **Policy Criteria** section, configure the attributes that you you want to trigger the policy for.

9. Click **Next**.

10. In the **Policy scope** section, configure the following:

    - Click **Add inclusion scope** to use **Restrict to Scope** to enable this policy only for a specific cluster, a namespace, or a label. You can add multiple scopes and also use regular expressions in RE2 Syntax for namespaces and labels.

    - Click **Add exclusion scope** to use **Exclude by Scope** to exclude deployments, clusters, namespaces, and labels you specify, it means that the policy will not apply to the entities that you select. You can add multiple scopes and also use regular expressions in RE2 Syntax for namespaces and labels. However, you cannot use regular expressions for selecting deployments.

    - For **Excluded Images (Build Lifecycle only)**, select all images that you do not want to trigger a violation for.

      > **NOTE**
      >
      > The **Excluded Images** setting only applies when you check images in a continuous integration system with the **Build** lifecycle stage. It does not have any effect if you use this policy to check running deployments in the **Deploy** lifecycle stage or runtime activities in the **Runtime** lifecycle stage.

11. Click **Next**.

12. In the **Review policy** section, preview the policy violations.

13. Click **Save**.

Additional resources

- Policy criteria

6.4.2. Creating a security policy from the risk view

While evaluating risks in your deployments in the **Risk** view, when you apply local page filtering, you can create new security policies based on the filtering criteria you are using.

Procedure

1. Navigate to the RHACS portal and select **Risk** from the navigation menu.

2. Apply local page filtering criteria that you want to create a policy for.

3. Select **New Policy** and fill in the required fields to create a new policy.

Additional resources

- Using local page filtering

- Creating a security policy from the system policies view

6.4.3. Policy criteria

In the **Policy Criteria** section you can configure the data on which you want to trigger a policy.

You can configure the policy based on the attributes listed in the following table.

In this table:

- The **Regular expressions**, **AND, OR**, and **NOT** columns indicate whether you can use regular expressions and other logical operators along with the specific attribute.

    - **!** in the **Regular expressions** column indicates that you can only use regular expressions for the listed fields.

    - **!** in the **AND, OR** column indicates that you can only use the mentioned logical operator for the attribute.

- The **RHACS version** column indicates the version of Red Hat Advanced Cluster Security for Kubernetes that you must have to use the attribute.

- You cannot use logical combination operators **AND** and **OR** for attributes that have:

    - Boolean values **true** and **false**

    - Minimum-value semantics, for example:

        - Minimum RBAC permissions

        - Days since image was created

- You cannot use the **NOT** logical operator for attributes that have:

    - Boolean values **true** and **false**

    - Numeric values that already use comparison, such as the **<**, **>**, **<=**, **>=** operators.

    - Compound criteria that can have multiple values, for example:

        - **Dockerfile Line**, which includes both instructions and arguments.

        - **Environment Variable**, which consists of both name and value.

    - Other meanings, including **Add Capabilities**, **Drop Capabilities**, **Days since image was created**, and **Days since image was last scanned**

NOTE

To use logical operators **AND**, **OR**, and **NOT** for creating security policies, you need Red Hat Advanced Cluster Security for Kubernetes version 3.0.45 or newer. However, on earlier versions you can still use regular expressions for the fields listed in the **Regular expressions** column.

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Namespace	The name of the namespace.	3.0.51 and newer	✓	✓	✓	Deploy
Image Registry	The name of the image registry.	All	✓	✓	✓	Deploy
Image Remote	The full name of the image in registry, for example **library/nginx**.	All	✓	✓	✓	Deploy
Image Tag	Identifier for an image.	All	✓	✓	✓	Deploy
Days since image was created	The number of days from image creation date.	All	✕	✕	✕	Build
Days since image was last scanned	The number of days since the last image scan.	All	✕	✕	✕	Build
Dockerfile Line	A specific line in the Dockerfile, including both instructions and arguments.	All	! only for values	✕	✓	Build

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Image is NOT Scanned	No scan data is available for the image.	All	✕	✕	✕	Build
CVSS	Common Vulnerability Scoring System, use it to match images with vulnerabilities whose scores are greater than **>**, less than **<**, or equal to **=** the specified CVSS.	All	✕	✕	✓	Build
Fixed By	The version string of a package that fixes a flagged vulnerability in an image.	All	✓	✓	✓	Build
CVE	Common Vulnerabilities and Exposures, use it with specific CVE numbers.	All	✓	✓	✓	Build
Image Component	Name and version number of a specific software component present in an image.	All	✓	✕	✓	Build

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Image OS	Name and version number of the base operating system of the image.	3.0.47 and newer	✓	✓	✓	Build
Environment Variable	Check environment variables by name or value.	All	! only for key and value	✕	✓	Deploy
Disallowed Annotation	An annotation which is not allowed to be present on Kubernetes resources in a specified environment.	All	✓	✕	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Disallowed Image Label	Check for the presence of a Docker image label that should not be in use. The policy triggers if any image in the deployment has the specified label. You can use regular expressions for both **key** and **value** fields to match labels. The **Disallowed Image Label** policy criteria only works when you integrate with a Docker registry.	3.0.40 and newer	✓	✗	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Required Image Label	Check for the presence of a required Docker image label. The policy triggers if any image in the deployment does not have the specified label. You can use regular expressions for both **key** and **value** fields to match labels. The **Required Image Label** policy criteria only works when you integrate with a Docker registry.	3.0.40 and newer	✓	✕	✓	Deploy
Required Label	Check for the presence of a required label in Kubernetes.	All	✓	✕	✓	Deploy
Required Annotation	Check for the presence of a required annotation in Kubernetes.	All	✓	✕	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Volume Name	Name of the storage.	All	✓	✓	✓	Deploy
Volume Source	Indicates the form in which the volume is provisioned. For example, **persistent VolumeClaim** or **hostPath**.	All	✓	✓	✓	Deploy
Volume Destination	The path where the volume is mounted.	All	✓	✓	✓	Deploy
Volume Type	The type of volume.	All	✓	✓	✓	Deploy
Writable Volume	Volumes that are mounted as writable.	All	✕	✕	✕	Deploy
Protocol	Protocol, such as, TCP or UDP, that is used by the exposed port.	All	✓	✓	✓	Deploy
Port	Port numbers exposed by a deployment.	All	✕	✓	✓	Deploy
Privileged	Privileged running deployments.	All	✕	✕	✕	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Read-Only Root Filesystem	Containers running with the root file system configured as read only.	All	✕	✕	✕	Deploy
Drop Capabilities	Linux capabilities that must be dropped from the container. For example **CAP_SET UID** or **CAP_NET _RAW**.	All	✕	✕	✓	Deploy
Add Capabilities	Linux capabilities that must not be added to the container, for instance the ability to send raw packets or override file permissions.	All	✕	✕	✓	Deploy
Process Name	Name of the process executed in a deployment.	All	✓	✓	✓	Runtime
Process Ancestor	Name of any parent process for a process executed in a deployment.	All	✓	✓	✓	Runtime

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Process Arguments	Command arguments for a process executed in a deployment.	All	✓	✓	✓	Runtime
Process UID	Unix user ID for a process executed in a deployment.	All	✗	✓	✓	Runtime
Port Exposure	Exposure method of the service, for example, load balancer or node port.	All	✗	✓	✓	Deploy
Service Account	The name of the service account.	All	✓	✓	✓	Deploy
Writable Host Mount	Resource has mounted a path on the host with write permissions.	All	✗	✗	✗	Deploy
Unexpected Process Executed	Check deployments for which process executions are not listed in the deployment's locked process baseline.	All	✗	✗	✗	Runtime

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Minimum RBAC Permissions	Match if the deployment's Kubernetes service account has Kubernetes RBAC permission level equal to **=** or greater than **>** the specified level.	All	✕	✓	✕	Deploy
Container Name	The name of the container.	3.0.52 and newer	✓	✓	✓	Deploy
Container CPU Request	Check for the number of cores reserved for a given resource.	All	✕	✕	✓	Deploy
Container CPU Limit	Check for the maximum number of cores a resource is allowed to use.	All	✕	✕	✓	Deploy
Container Memory Request	Check for the amount of memory reserved for a given resource.	All	✕	✕	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Container Memory Limit	Check for the maximum amount of memory a resource is allowed to use.	All	✕	✕	✓	Deploy
Kubernetes Action	The name of the Kubernetes action, such as **Pod Exec**.	3.0.55 and newer	✕	✕	! **OR** only	Runtime
Kubernetes Resource	The name of the accessed Kubernetes resource, such as **configmap s** or **secrets**.	3.63 and newer	✕	✕	! **OR** only	Runtime
Kubernetes Resource Name	The name of the accessed Kubernetes resource.	3.63 and newer	✓	✓	! **OR** only	Runtime
Kubernetes API Verb	The Kubernetes API verb that is used to access the resource, such as **GET** or **POST**.	3.63 and newer	✕	✕	! **OR** only	Runtime
Kubernetes User Name	The name of the user who accessed the resource.	3.63 and newer	✓	✓	! **OR** only	Runtime

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Kubernetes User Group	The name of the group to which the user who accessed the resource belongs to.	3.63 and newer	✓	✕	! **OR** only	Runtime
User Agent	The user agent that the user used to access the resource. For example **oc**, or **kubectl**.	3.63 and newer	✓	✓	! **OR** only	Runtime
Source IP Address	The IP address from which the user accessed the resource.	3.63 and newer	✓	✓	! **OR** only	Runtime
Is Impersonated User	Check if the request was made by a user that is impersonated by a service account or some other account.	3.63 and newer	✕	✕	✕	Runtime
Runtime Class	The RuntimeClass of the deployment.	3.67 and newer	✓	✓	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Automount Service Account Token	Check if the deployment configuration automatically mounts the service account token.	3.68 and newer	✕	✕	✕	Deploy
Liveness Probe	Whether the container defines a liveness probe.	3.69 and newer	✕	✕	✕	Deploy
Readiness Probe	Whether the container defines a readiness probe.	3.69 and newer	✕	✕	✕	Deploy
Replicas	The number of deployment replicas.	3.69 and newer	✕	✓	✓	Deploy
Privilege escalation	Provides alerts when a development is configured to allow a container process to gain more privileges than its parent process.	3.70 and later	✕	✕	✕	Deploy
Ingress Network Policy	Check the presence or absence of ingress Kubernetes network policies.	3.70 and later	✕	✕	✓	Deploy

Attribute	Description	RHACS version	Regular expressions	NOT	AND, OR	Phase
Egress Network Policy	Check the presence or absence of egress Kubernetes network policies.	3.70 and later	✕	✕	✓	Deploy
Not verified by trusted image signers	The list of signature integrations you can use to verify an image's signature. Create alerts on images that either do not have a signature or their signature is not verifiable by at least one of the provided signature integrations.	3.70 and later	✕	✕	! **OR** only	Deploy
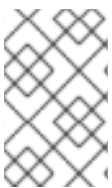
> **NOTE**
>
> If you are using Red Hat Advanced Cluster Security for Kubernetes version 3.0.44 or older, the policy criteria you specify in the **Policy criteria** section are "AND"ed. It means that the violation only triggers if all the specified policy criteria match.

6.4.3.1. Adding logical conditions for the policy criteria

You can use the drag-and-drop policy fields panel to specify logical conditions for the policy criteria.

Prerequisites

- You must be using Red Hat Advanced Cluster Security for Kubernetes version 3.0.45 or newer.

Procedure

1. In the **Policy Criteria** section, select **Add a new condition** to add a new policy section.

- You can click on the **Edit** icon to rename the policy section.

- The **Drag out a policy field**section lists available policy criteria in multiple categories. You can expand and collapse these categories to view the policy criteria attributes.

2. Drag an attribute to the **Drop a policy field inside**area of the policy section.

3. Depending on the type of the attribute you select, you get different options to configure the conditions for the selected attribute. For example:

   - If you select an attribute with Boolean values **Read-Only Root Filesystem**, you will see **READ-ONLY** and **WRITABLE** options.

   - If you select an attribute with compound values **Environment variable**, you will see options to enter values for **Key**, **Value**, and **Value From** fields, and an icon to add more values for the available options.

     a. To combine multiple values for an attribute, click the **Add** icon.

     b. You can also click on the logical operator **AND** or **OR** listed in a policy section, to toggle between **AND** and **OR** operators. Toggling between operators only works inside a policy section and not between two different policy sections.

4. You can specify more than one **AND** and **OR** condition by repeating these steps. After you configure the conditions for the added attributes, click **Next** to continue with the policy creation.

6.5. SHARING SECURITY POLICIES

Beginning from Red Hat Advanced Cluster Security for Kubernetes version 3.0.44, you can share your security policies between different Central instances, by exporting and importing policies. It helps you enforce the same standards for all your clusters. To share policies, you export them as JSON files, and then import them back into another Central instance.
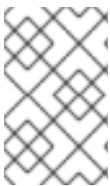
> **NOTE**
>
> Currently, you cannot export multiple security policies at once by using the RHACS portal. However, you can use the API for exporting multiple security policies. On the RHACS portal, navigate to **Help → API reference** to see the API reference.

6.5.1. Exporting a security policy

When you export a policy, it includes all the policy contents and also includes cluster scopes, cluster exclusions, and all configured notifications.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. From the **Policies** page, select the policy you want to edit.

3. Select **Actions → Export policy to JSON**

6.5.2. Importing a security policy

You can import a security policy from the **System Policies** view on the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Policies**.

2. Click **Import policy**.

3. In the **Import policy JSON** dialog, click **Upload** and select the JSON file you want to upload.

4. Click **Begin import**.

Each security policy in Red Hat Advanced Cluster Security for Kubernetes has a unique ID (UID) and a unique name. When you import a policy, Red Hat Advanced Cluster Security for Kubernetes handles the uploaded policy as follows:

- If the imported policy UID and name do not match any existing policy, Red Hat Advanced Cluster Security for Kubernetes creates a new policy.

- If the imported policy has the same UID as an existing policy, but a different name, you can either:

  - Keep both policies. Red Hat Advanced Cluster Security for Kubernetes saves the imported policy with a new UID.

  - Replace the existing policy with the imported policy.

- If the imported policy has the same name as an existing policy, but a different UID, you can either:

  - Keep both policies by providing a new name for the imported policy.

  - Replace the existing policy with the imported policy.

- If the imported policy has the same name and UID as an existing policy, the Red Hat Advanced Cluster Security for Kubernetes checks if the policy criteria match to the existing policy. If the policy criteria match, Red Hat Advanced Cluster Security for Kubernetes keeps the existing policy and shows a success message. If the policy criteria do not match, you can either:

  - Keep both policies by providing a new name for the imported policy.

  - Replace the existing policy with the imported policy.
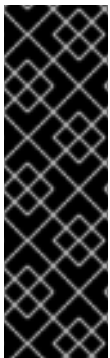
> **IMPORTANT**
>
> - If you import into the same Central instance, Red Hat Advanced Cluster Security for Kubernetes uses all the exported fields.
>
> - If you import into a different Central instance, Red Hat Advanced Cluster Security for Kubernetes omits certain fields, such as cluster scopes, cluster exclusions, and notifications. Red Hat Advanced Cluster Security for Kubernetes shows these omitted fields in a message. These fields vary for every installation, and you cannot migrate them from one Central instance to another.

CHAPTER 7. DEFAULT SECURITY POLICIES

The default security policies in Red Hat Advanced Cluster Security for Kubernetes provide broad coverage to identify security issues and ensure best practices for security in your environment. By configuring those policies, you can automatically prevent high-risk service deployments in your environment and respond to runtime security incidents.

> **NOTE**
>
> The severity levels for policies in Red Hat Advanced Cluster Security for Kubernetes are different from the severity levels that Red Hat Product Security assigns.
>
> The Red Hat Advanced Cluster Security for Kubernetes policy severity levels are Critical, High, Medium, and Low. Red Hat Product Security rates vulnerability severity levels as Critical, Important, Moderate, and Low.
>
> While a policy's severity level and the Red Hat Product Security severity levels can interact, it is important to distinguish between them. For more information about the Red Hat Product Security severity levels, see Severity Ratings.

7.1. CRITICAL SEVERITY SECURITY POLICIES

The following table lists the default security policies in Red Hat Advanced Cluster Security for Kubernetes that are of critical severity. The policies are organized by life cycle stage.

Table 7.1. Critical severity security policies

Life cycle stage	Name	Description	Status
Build or Deploy	Apache Struts: CVE-2017-5638	Alerts when deployments have images that contain the CVE-2017-5638 Apache Struts vulnerability.	Enabled
Build or Deploy	Log4Shell: log4j Remote Code Execution vulnerability	Alerts when deployments include images that contain the CVE-2021-44228 and CVE-2021-45046 Log4Shell vulnerabilities. Flaws exist in the Apache Log4j Java logging library in versions 2.0-beta9 – 2.15.0, excluding version 2.12.2.	Enabled

Life cycle stage	Name	Description	Status
Build or Deploy	Spring4Shell (Spring Framework Remote Code Execution) and Spring Cloud Function vulnerabilities	Alerts when deployments include images that contain either the CVE-2022-22965 vulnerability, which affects Spring MVC, and the CVE-2022-22963 vulnerability, which affects Spring Cloud. In versions 3.16, 3.2.2, and older unsupported versions, Spring Cloud contains flaws. Flaws exist in Spring Framework in versions 5.3.0 - 5.3.17, versions 5.2.0 - 5.2.19, and in older unsupported versions.	Enabled
Runtime	Iptables Executed in Privileged Container	Alerts when privileged pods run iptables.	Enabled

7.2. HIGH SEVERITY SECURITY POLICIES

The following table lists the default security policies in Red Hat Advanced Cluster Security for Kubernetes that are of high severity. The policies are organized by life cycle stage.

Table 7.2. High severity security policies

Life cycle stage	Name	Description	Status
Build or Deploy	Fixable CVSS >= 7	Alerts when deployments with fixable vulnerabilities have a CVSS of at least 7.	Disabled
Build or Deploy	Fixable Severity at least Important	Alerts when deployments with fixable vulnerabilities have a severity rating of at least Important.	Enabled

Life cycle stage	Name	Description	Status
Build or Deploy	Secure Shell (ssh) Port Exposed in Image	Alerts when deployments expose port 22, which is commonly reserved for SSH access.	Enabled
Deploy	Emergency Deployment Annotation	Alerts when deployments use the emergency annotation, such as "admission.stackrox.io/break-glass":"ticket-1234" to circumvent StackRox Admission controller checks.	Enabled
Deploy	Environment Variable Contains Secret	Alerts when deployments have environment variables that contain 'SECRET'.	Enabled
Deploy	Fixable CVSS >= 6 and Privileged	Alerts when deployments run in privileged mode with fixable vulnerabilities that have a CVSS of at least 6.	Disabled by default in version 3.72.0 and later
Deploy	Privileged Containers with Important and Critical Fixable CVEs	Alerts when containers that run in privileged mode have important or critical fixable vulnerabilities.	Enabled
Deploy	Secret Mounted as Environment Variable	Alerts when a deployment has a Kubernetes secret that is mounted as an environment variable.	Disabled
Deploy	Secure Shell (ssh) Port Exposed	Alerts when deployments expose port 22, which is commonly reserved for SSH access.	Enabled
Runtime	Cryptocurrency Mining Process Execution	Spawns the crypto-currency mining process.	Enabled

Life cycle stage	Name	Description	Status
Runtime	iptables Execution	Detects when someone runs iptables, which is a deprecated way of managing network states in containers.	Enabled
Runtime	Kubernetes Actions: Exec into Pod	Alerts when the Kubernetes API receives a request to run a command in a container.	Enabled
Runtime	Linux Group Add Execution	Detects when someone runs the addgroup or groupadd binary to add a Linux group.	Enabled
Runtime	Linux User Add Execution	Detects when someone runs the useradd or adduser binary to add a Linux user.	Enabled
Runtime	Login Binaries	Indicates when someone tries to log in.	Disabled
Runtime	Network Management Execution	Detects when someone runs binary files that can manipulate network configuration and management.	Enabled
Runtime	nmap Execution	Alerts when someone starts the nmap process in a container during run time.	Enabled
Runtime	OpenShift: Kubeadmin Secret Accessed	Alerts when someone accesses the kubeadmin secret.	Enabled
Runtime	Password Binaries	Indicates when someone attempts to change a password.	Disabled
Runtime	Process Targeting Cluster Kubelet Endpoint	Detects the misuse of the healthz, kubelet API, or heapster endpoint.	Enabled

Life cycle stage	Name	Description	Status
Runtime	Process Targeting Cluster Kubernetes Docker Stats Endpoint	Detects the misuse of the Kubernetes docker stats endpoint.	Enabled
Runtime	Process Targeting Kubernetes Service Endpoint	Detects the misuse of the Kubernetes Service API endpoint.	Enabled
Runtime	Process with UID 0	Alerts when deployments contain processes that run with UID 0.	Disabled
Runtime	Secure Shell Server (sshd) Execution	Detects containers that run the SSH daemon.	Enabled
Runtime	SetUID Processes	Use setuid binary files, which permit people to run certain programs with escalated privileges.	Disabled
Runtime	Shadow File Modification	Indicates when someone tries to modify shadow files.	Disabled
Runtime	Shell Spawned by Java Application	Detects when a shell, such as bash, csh, sh, or zsh, is run as a subprocess of a Java application.	Enabled
Runtime	Unauthorized Network Flow	Generates a violation for any network flows that fall outside of the baselines of the "alert on anomalous violations" setting.	Enabled
Runtime	Unauthorized Processed Execution	Generates a violation for any process execution that is not explicitly allowed by a locked process baseline for a container specification in a Kubernetes deployment.	Enabled

7.3. MEDIUM SEVERITY SECURITY POLICIES

The following table lists the default security policies in Red Hat Advanced Cluster Security for Kubernetes that are of medium severity. The policies are organized by life cycle stage.

Table 7.3. Medium severity security policies

Life cycle stage	Name	Description	Status
Build	Docker CIS 4.4: Ensure images are scanned and rebuilt to include security patches	Alerts when images are not scanned and rebuilt to include security patches. It is important to scan images often to find vulnerabilities, rebuild the images to include security patches, and then instantiate containers for the images.	Disabled
Deploy	30-Day Scan Age	Alerts when a deployment has not been scanned in 30 days.	Enabled
Deploy	CAP_SYS_ADMIN capability added	Alerts when a deployment includes containers that are escalating with CAP_SYS_ADMIN.	Enabled
Deploy	Container using read-write root filesystem	Alerts when a deployment includes containers that have read-write root file systems.	Disabled
Deploy	Container with privilege escalation allowed	Alerts when a container might be running with unintended privileges, creating a security risk. This situation can happen when a container process that has more privileges than its parent process allows the container to run with unintended privileges.	Enabled
Deploy	Deployments should have at least one Ingress Network Policy	Alerts if deployments are missing an Ingress Network Policy.	Disabled

Life cycle stage	Name	Description	Status
Deploy	Deployments with externally exposed endpoints	Detects if a deployment has any service that is externally exposed through any methods. Deployments with services exposed outside of the cluster are at a higher risk of attempted intrusions because they are reachable outside of the cluster. This policy provides an alert so that you can verify that service exposure outside of the cluster is required. If the service is only needed for intra-cluster communication, use service type ClusterIP.	Disabled
Deploy	Docker CIS 5.1: Ensure that, if applicable, an AppArmor profile is enabled	Uses AppArmor to protect the Linux operating system and applications by enforcing a security policy that is known as an AppArmor profile. AppArmor is a Linux application security system that is available on some Linux distributions by default, such as Debian and Ubuntu.	Enabled
Deploy	Docker CIS 5.15: Ensure that the host's process namespace is not shared	Creates process-level isolation between the containers and the host. The Process ID (PID) namespace isolates the process ID space, which means that processes in different PID namespaces can have the same PID.	Enabled

Life cycle stage	Name	Description	Status
Deploy	Docker CIS 5.16: Ensure that the host's IPC namespace is not shared	Alerts when the IPC namespace on the host is shared with containers. The IPC (POSIX/SysV IPC) namespace separates named shared memory segments, semaphores, and message queues.	Enabled
Deploy	Docker CIS 5.19: Ensure mount propagation mode is not enabled	Alerts when mount propagation mode is enabled. When mount propagation mode is enabled, you can mount container volumes in Bidirectional, Host to Container, and None modes. Do not use Bidirectional mount propagation mode unless it is explicitly needed.	Enabled
Deploy	Docker CIS 5.21: Ensure the default seccomp profile is not disabled	Alerts when the seccomp profile is disabled. The seccomp profile uses an allowlist to permit common system calls and blocks all others.	Disabled
Deploy	Docker CIS 5.7: Ensure privileged ports are not mapped within containers	Alerts when privileged ports are mapped within containers. The TCP/IP port numbers that are lower than 1024 are privileged ports. Normal users and processes can not use them for security reasons, but containers might map their ports to privileged ports.	Enabled

Life cycle stage	Name	Description	Status
Deploy	Docker CIS 5.9 and 5.20: Ensure that the host's network namespace is not shared	Alerts when the host's network namespace is shared. When HostNetwork is enabled, the container is not placed inside a separate network stack, and the container's networking is not containerized. As a result, the container has full access to the host's network interfaces, and a shared UTS namespace is enabled. The UTS namespace provides isolation between the hostname and the NIS domain name, and it sets the hostname and the domain, which are visible to running processes in that namespace. Processes that run within containers do not typically require to know the hostname or the domain name, so the UTS namespace should not be shared with the host.	Enabled
Deploy	Images with no scans	Alerts when a deployment includes images that were not scanned.	Disabled
Runtime	Kubernetes Actions: Port Forward to Pod	Alerts when the Kubernetes API receives a port forward request.	Enabled
Deploy	Mount Container Runtime Socket	Alerts when a deployment has a volume mount on the container runtime socket.	Enabled

Life cycle stage	Name	Description	Status
Deploy	Mounting Sensitive Host Directories	Alerts when a deployment mounts sensitive host directories.	Enabled
Deploy	No resource requests or limits specified	Alerts when a deployment includes containers that do not have resource requests and limits.	Enabled
Deploy	Pod Service Account Token Automatically Mounted	Protects pod default service account tokens from being compromised by minimizing the mounting of the default service account token to only those pods whose applications require interaction with the Kubernetes API.	Enabled
Deploy	Privileged Container	Alerts when a deployment includes containers that run in privileged mode.	Enabled
Runtime	crontab Execution	Detects the usage of the crontab scheduled jobs editor.	Enabled
Runtime	Netcat Execution Detected	Detects when netcat runs in a container.	Enabled
Runtime	OpenShift: Advanced Cluster Security Central Admin Secret Accessed	Alerts when someone accesses the Red Hat Advanced Cluster Security Central secret.	Enabled
Runtime	OpenShift: Kubernetes Secret Accessed by an Impersonated User	Alerts when someone impersonates a user to access a secret in the cluster.	Enabled
Runtime	Remote File Copy Binary Execution	Alerts when a deployment runs a remote file copy tool.	Enabled

7.4. LOW SEVERITY SECURITY POLICIES

The following table lists the default security policies in Red Hat Advanced Cluster Security for Kubernetes that are of low severity. The policies are organized by life cycle stage.

Table 7.4. Low severity security policies

Life cycle stage	Name	Description	Status
Build or Deploy	90-Day Image Age	Alerts when a deployment has not been updated in 90 days.	Enabled
Build or Deploy	ADD Command used instead of COPY	Alerts when a deployment uses an ADD command.	Disabled
Build or Deploy	Alpine Linux Package Manager (apk) in Image	Alerts when a deployment includes the Alpine Linux package manager (apk).	Enabled
Build or Deploy	Curl in Image	Alerts when a deployment includes curl.	Disabled
Build or Deploy	Docker CIS 4.1: Ensure That a User for the Container Has Been Created	Ensures that containers are running as non-root users.	Enabled
Build or Deploy	Docker CIS 4.7: Alert on Update Instruction	Ensures that update instructions are not used alone in the Dockerfile.	Enabled
Build or Deploy	Insecure specified in CMD	Alerts when a deployment uses 'insecure' in the command.	Enabled
Build or Deploy	Latest tag	Alerts when a deployment includes images that use the 'latest' tag.	Enabled
Build or Deploy	Red Hat Package Manager in Image	Alerts when a deployment includes components of the Red Hat, Fedora, or CentOS package management system.	Enabled

Life cycle stage	Name	Description	Status
Build or Deploy	Required Image Label	Alerts when a deployment includes images that are missing the specified label.	Disabled
Build or Deploy	Ubuntu Package Manager in Image	Alerts when a deployment includes components of the Debian or Ubuntu package management system in the image.	Enabled
Build or Deploy	Wget in Image	Alerts when a deployment includes wget.	Disabled
Deploy	Improper Usage of Orchestrator Secrets Volume	Alerts when a deployment uses a Dockerfile with 'VOLUME /run/secrets'.	Enabled
Deploy	Kubernetes Dashboard Deployed	Alerts when a Kubernetes dashboard service is detected.	Enabled
Deploy	Required Annotation: Email	Alerts when a deployment is missing the 'email' annotation.	Disabled
Deploy	Required Annotation: Owner/Team	Alerts when a deployment is missing the 'owner' or 'team' annotation.	Disabled
Deploy	Required Label: Owner/Team	Alerts when a deployment is missing the 'owner' or 'team' label.	Disabled
Runtime	Alpine Linux Package Manager Execution	Alerts when the Alpine Linux package manager (apk) is run at run time.	Enabled
Runtime	chkconfig Execution	Detects the usage of the ckconfig service manager, which is typically not used in a container.	Enabled

Life cycle stage	Name	Description	Status
Runtime	Compiler Tool Execution	Alerts when binary files that compile software are run at run time.	Enabled
Runtime	Red Hat Package Manager Execution	Alerts when Red Hat, Fedora, or CentOS package manager programs are run at run time.	Enabled
Runtime	Shell Management	Alerts when commands are run to add or remove a shell.	Disabled
Runtime	systemctl Execution	Detects the usage of the systemctl service manager.	Enabled
Runtime	systemd Execution	Detects the usage of the systemd service manager.	Enabled

CHAPTER 8. MANAGING NETWORK POLICIES

A Kubernetes network policy is a specification of how groups of pods are allowed to communicate with each other and other network endpoints. These network policies are configured as YAML files. By looking at these files alone, it is often hard to identify whether the applied network policies achieve the desired network topology.

Red Hat Advanced Cluster Security for Kubernetes gathers all defined network policies from your orchestrator and provides functionality to make these policies easier to use.

To support network policy enforcement, Red Hat Advanced Cluster Security for Kubernetes provides:

- Network graph

- Network policy simulator

- Network policy generator

- Build-time network policy generator

8.1. NETWORK GRAPH VIEW

The network graph provides visibility and control over:

- The allowed network connections, as defined by the Kubernetes network policies.

- The active communications paths among namespaces and deployments.

In the menu bar, choose the cluster for which you want to view information and at least one namespace.

In the **Network Graph** view, you can configure the type of connections you want to see. In the **Flows** section (upper left), select:

- **Active** to view only active connections.

- **Allowed** to view only allowed network connections.

- **All** to view both active and allowed network connections.

In the **Network Graph**, click **Legend** to view information about the symbols in use and their meaning. The legend shows explanatory text for symbols representing namespaces, deployments, and connections on the network graph.

Clicking on an item in the **Network Graph**, such as a deployment or a namespace, opens a window that displays additional information. You can expand and collapse the window by selecting the arrow in the blue bar.

When you hover over:

- a connection, you see information about the network flow, which includes active connections, port numbers, and protocols in use.

- a deployment, you see information about ingress and egress connections, protocols, port numbers in use, and the direction of the network traffic between deployments.

Allowed network connections

Red Hat Advanced Cluster Security for Kubernetes processes all network policies in each secured cluster to show you which deployments can contact each other and which can reach external networks.

The network graph shows possible network connections as dashed lines.

Actual network flows

Red Hat Advanced Cluster Security for Kubernetes monitors running deployments and tracks traffic between them. The network graph shows observed network flows as solid lines.

Network baseline

Red Hat Advanced Cluster Security for Kubernetes discovers existing network flows and creates a baseline.

To view the network baseline for a deployment, select that deployment in the **Network Graph** view. The **Network Flows** details panel show both anomalous and baseline flows. From this panel, you can:

- Mark network flows from the baseline as anomalous by selecting **Mark as Anomalous**.

- Add network flows to the baseline from the anomalous flows by selecting **Add to Baseline**.

External entities and connections

The **Network Graph** view shows network connections between managed clusters and external sources. In addition, Red Hat Advanced Cluster Security for Kubernetes automatically discovers and highlights public Classless Inter-Domain Routing (CIDR) address blocks, such as Google Cloud, AWS, Azure, Oracle Cloud, and Cloudflare. Using this information, you can identify deployments with active external connections and determine if they are making or receiving unauthorized connections from outside your network.

By default, the external connections point to a common **External Entities** box and different CIDR address blocks in the **Network Graph** view. However, you can choose not to show auto-discovered CIDR blocks.

Red Hat Advanced Cluster Security for Kubernetes includes IP ranges for the following cloud providers:

- Google Cloud

- AWS

- Azure

- Oracle Cloud

- Cloudflare

Red Hat Advanced Cluster Security for Kubernetes fetches and updates the cloud providers' IP ranges every 7 days. If you are using offline mode, you can update these ranges by installing new support packages.

Additional resources

- Updating kernel support packages in offline mode

8.1.1. Viewing network policies

Network policies specify how groups of pods are allowed to communicate with each other and with other network endpoints. Kubernetes **NetworkPolicy** resources use labels to select pods and define

rules that specify what traffic is allowed to or from the selected pods. Red Hat Advanced Cluster Security for Kubernetes discovers and displays network policy information for all your Kubernetes clusters, namespaces, deployments, and pods, in the **Network Graph** view.

To view network policies and other related details for deployments in a namespace, you can select the namespace in **Network Graph** view.

The namespace details panel lists all deployments in the selected namespace. You can then hover over a deployment in the details panel and select the **Navigate to deployment** (arrow) icon that appears on the right to view deployment details.

Alternatively, you can directly select a deployment in the **Network Graph** view to view its details. The deployment details panel includes the **Network Flows**, **Details**, and **Network Policies** tabs.

You can select each tab to view related information.

- The **Network Flows** tab shows information about ingress and egress connections, protocols, and port numbers in use for that deployment.

- The **Details** tab shows information about how the service is deployed, including orchestrator labels and annotations.

- The **Network Policies** tab shows information about every network policy that applies to the deployment.

> **NOTE**
>
> You need Red Hat Advanced Cluster Security for Kubernetes version 3.0.47 or newer to see information about ingress and egress connections, protocols, port numbers, and the direction of the network traffic.

8.1.2. Configuring CIDR blocks

You can specify custom CIDR blocks or configure displaying auto-discovered CIDR block in the **Network Graph** view.

Procedure

1. In the **Network Graph** view, select **Configure CIDR Blocks**.

2. Turn the **Display auto-discovered CIDR blocks in Network Graph** toggle off to hide auto-discovered CIDR blocks.

> **NOTE**
>
> When you hide the auto-discovered CIDR blocks, the auto-discovered CIDR blocks are hidden for all clusters, not only for the selected cluster on the top bar in the **Network Graph** view.

3. Add custom CIDR addresses by adding **CIDR Block Name** and **CIDR Address**. To add more than one, select the **Add** icon.

4. Click **Update Configuration** to save the changes.

8.2. SIMULATING NETWORK POLICIES

Your current network policies might allow unneeded network communications. To simulate the effects of a new set of network policies, use the network policy simulator.

Procedure

1. On the RHACS portal, select **Network Graph** from the left-hand navigation menu.

2. Select one or more namespaces.

3. On the **Network Graph** view header, select **Network Policy Simulator**.

4. Select **Upload and simulate network policy YAML** and upload your proposed YAML file. The network graph view displays what your proposed network policies would achieve.

5. To share your proposed policies with your team, select **Share YAML**. To apply your policy directly, select **Apply Network Policies**.

> **WARNING**
>
> Directly applying network policies might cause problems for running applications. Always download and test the network policies in a development environment or test clusters before applying them to production workloads.

8.3. GENERATING NETWORK POLICIES

A Kubernetes network policy controls which pods receive incoming network traffic, and which pods can send outgoing traffic. By using network policies to enable and disable traffic to or from pods, you can limit your network attack surface.

These network policies are YAML configuration files. It is often difficult to gain insights into the network flow and manually create these files. Red Hat Advanced Cluster Security for Kubernetes allows you to autogenerate these network policies based on the actual observed network communication flows in your environment.

You can generate network policies from the network graph view.

The generated policies apply to the deployments shown in the network graph and they allow all network traffic observed during the selected time.

Procedure

1. In the RHACS portal, select **Network Graph** from the left-hand navigation menu.

2. Select a cluster name from the menu on the top bar if the correct one is not already selected.

3. Select one or more namespaces.

4. If you want to generate policies for only some deployments, use the **Add one or more deployment filters** field to add criteria by which to filter deployments. If you do not add a filter,

Red Hat Advanced Cluster Security for Kubernetes generates policies for all deployments in the cluster.

5. Select an appropriate time from the menu on the top bar. If the selected time is too short it leaves out periodic or infrequent network communications.

6. Select **Network Policy Simulator**.

7. In the panel that opens, select **Exclude ports & protocols** if you do not want ports and protocols to be scoped in Red Hat Advanced Cluster Security for Kubernetes generated policies.

8. Select **Generate and simulate network policies**. The generated network policy configuration YAML opens in the same panel, and the network graph shows the effects of the policies.

8.3.1. Saving generated policies

You can download and save the generated network policies from Red Hat Advanced Cluster Security for Kubernetes. Use this option to commit the policies into a version control system like Git.

Procedure

- Select the **Download YAML** icon on the **Network Policy Simulator** panel.

8.3.2. Testing generated policies

After you download the network policies that Red Hat Advanced Cluster Security for Kubernetes generates, you can test them by applying them to your cluster.

Procedure

1. To create policies using the saved YAML file, use the following command:

   ```
   $ oc create -f "<generated_file>.yaml" ❶
   ```

   ❶   If you use Kubernetes, enter **kubectl** instead of **oc**.

2. If the generated policies cause problems, you can remove them by running the following command:

   ```
   $ oc delete -f "<generated_file>.yaml" ❶
   ```

   ❶   If you use Kubernetes, enter **kubectl** instead of **oc**.

8.3.3. Applying generated policies

You can apply the generated network policies from the RHACS portal.

Procedure

- To directly apply the generated policies in the cluster from within Red Hat Advanced Cluster Security for Kubernetes, select **Apply Network Policies**.

> **WARNING**
>
> Directly applying network policies might cause problems for running applications. Always download and test the network policies in a development environment or test clusters before applying them to production workloads.

8.3.4. Deleting generated policies

If you have applied generated policies directly and want to remove them, select the **Revert most recently applied YAML** icon on the **Network Policy Simulator** panel.

Procedure

1. In the RHACS portal, select **Network Graph** from the left-hand navigation menu.

2. Select a cluster name from the menu on the top bar if the correct one is not already selected.

3. Select one or more namespaces.

4. Select **Network Policy Simulator**.

5. Select **View active YAMLS**.

6. Select the **Revert most recently applied YAML** icon.

8.3.5. Deleting all autogenerated policies

You can delete all generated policies from your cluster that you have created by using Red Hat Advanced Cluster Security for Kubernetes.

Procedure

- Run the following command:

  - On OpenShift Container Platform:

    ```
    $ oc get ns -o jsonpath='{.items[*].metadata.name}' | xargs -n 1 oc delete networkpolicies -l 'network-policy-generator.stackrox.io/generated=true' -n
    ```

  - On Kubernetes:

    ```
    $ kubectl get ns -o jsonpath='{.items[*].metadata.name}' | xargs -n 1 kubectl delete networkpolicies -l 'network-policy-generator.stackrox.io/generated=true' -n
    ```

8.3.6. Policy generation strategy

When you autogenerate network policies:

- Red Hat Advanced Cluster Security for Kubernetes generates a single network policy for each deployment in the namespace. The pod selector for the policy is the pod selector of the deployment.

  - If a deployment already has a network policy, Red Hat Advanced Cluster Security for Kubernetes does not generate new policies or delete existing policies.

- Generated policies only restrict traffic to existing deployments.

  - Deployments you create later will not have any restrictions unless you create or generate new network policies for them.

  - If a new deployment needs to contact a deployment with a network policy, you might need to edit the network policy to allow access.

- Each policy has the same name as the deployment name, prefixed with **stackrox-generated-**. For example, the policy name for the deployment **depABC** in the generated network policy is **stackrox-generated-depABC**. All generated policies also have an identifying label.

- Red Hat Advanced Cluster Security for Kubernetes generates a single rule allowing traffic from any IP address if:

  - The deployment has an incoming connection from outside the cluster within the selected time, or

  - The deployment is exposed through a node port or load balancer service.

- Red Hat Advanced Cluster Security for Kubernetes generates one **ingress** rule for every deployment from which there is an incoming connection.

  - For deployments in the same namespace, this rule uses the pod selector labels from the other deployment.

  - For deployments in different namespaces, this rule uses a namespace selector. To make this possible, Red Hat Advanced Cluster Security for Kubernetes automatically adds a label, **namespace.metadata.stackrox.io**/**name**, to each namespace.

> **IMPORTANT**
>
> In rare cases, if a standalone pod does not have any labels, the generated policy allows traffic from or to the pod's entire namespace.

8.4. USING BUILD-TIME NETWORK POLICY GENERATOR

> **IMPORTANT**
>
> Build-time network policy generation is a Technology Preview feature only. Technology Preview features are not supported with Red Hat production service level agreements (SLAs) and might not be functionally complete. Red Hat does not recommend using them in production. These features provide early access to upcoming product features, enabling customers to test functionality and provide feedback during the development process.
>
> For more information about the support scope of Red Hat Technology Preview features, see Technology Preview Features Support Scope .

The build-time network policy generator can automatically generate Kubernetes network policies based on application YAML manifests. You can use it to develop network policies as part of the CI/CD pipeline before deploying applications on your cluster.

Red Hat developed this feature in partnership with the developers of the NP-Guard project. First, the build-time network policy generator analyzes Kubernetes manifests in a local folder, including service manifests, config maps, and workload manifests such as **Pod**, **Deployment**, **ReplicaSet**, **Job**, **DaemonSet**, and **StatefulSet**. Then, it discovers the required connectivity and creates the Kubernetes network policies to achieve pod isolation. These policies allow no more and no less than the needed ingress and egress traffic.

8.4.1. Generating build-time network policies

The build-time network policy generator is included in the **roxctl** CLI. For the build-time network policy generation feature, **roxctl** CLI does not need to communicate with RHACS Central so you can use it in any development environment.

Prerequisites

1. The build-time network policy generator recursively scans the directory you specify when you run the command. Therefore, before you run the command, you must already have service manifests, config maps, and workload manifests such as **Pod**, **Deployment**, **ReplicaSet**, **Job**, **DaemonSet**, and **StatefulSet** as YAML files in the specified directory.

2. Verify that you can apply these YAML files as-is using the **kubectl apply -f** command. The build-time network policy generator does not work with files that use Helm style templating.

3. Verify that the service network addresses are not hard-coded. Every workload that needs to connect to a service must specify the service network address as a variable. You can specify this variable by using the workload's resource environment variable or in a config map.

   - Example 1: using an environment variable .

   - Example 2: using a config map

   - Example 3: using a config map

4. Service network addresses must match the following official regular expression pattern:

   ```
   (http(s)?://)?<svc>(.<ns>(.svc.cluster.local)?)?(:<portNum>)?  ❶
   ```

   ❶ In this pattern,

      - <svc> is the service name.

      - <ns> is the namespace where you defined the service.

      - <portNum> is the exposed service port number.

   Following are some examples that match the pattern:

   - **wordpress-mysql:3306**

   - **redis-follower.redis.svc.cluster.local:6379**

- **redis-leader.redis**

- **http://rating-service.**

Procedure

1. Verify that the build-time network policy generation feature is available by running the help command:

   ```
   $ roxctl generate netpol -h
   ```

2. Generate the policies by using the **generate netpol** command:

   ```
   $ roxctl generate netpol <folder-path> ❶
   ```

   ❶ Specify the path of the folder that has the Kubernetes manifests.

The **roxctl generate netpol** command supports the following options:

Option	Description
-h, --help	View the help text for the **netpol** command.
-d, --output-dir <dir>	Save the generated policies into a target folder. One file per policy.
-f, --output-file <filename>	Save and merge the generated policies into a single YAML file.
--fail	Fail on the first encountered error. The default value is **false**.
--remove	Remove the output path if it already exist.
--strict	Treat warnings as errors. The default value is **false**.

After generating the policies, you must inspect them for completeness and accuracy, in case any relevant network address was not specified as expected in the YAML files. Most importantly, verify that required connections are not blocked by the isolating policies. To help with this inspection, you can use RHACS to simulate the generated network policies .

> **NOTE**
>
> Red Hat recommends applying network policies to the cluster as part of the workload deployment using automation. You can follow a GitOps approach by submitting the generated policies using Pull Requests, providing the team an opportunity to review the policies before deploying them as part of the pipeline.

8.5. USING NETWORK BASELINING

In Red Hat Advanced Cluster Security for Kubernetes, you can minimize your risks by using network baselining. It is a proactive approach to keep your infrastructure secure. Red Hat Advanced Cluster Security for Kubernetes first discovers existing network flows and creates a baseline, and then it treats

network flows outside of this baseline as anomalous.

> **NOTE**
>
> - To use the **Network baseline** feature, you must use the Red Hat Advanced Cluster Security for Kubernetes version 3.0.54 or newer.
>
> - To enable alerts on baseline violations, you must use Red Hat Advanced Cluster Security for Kubernetes version 3.0.56 or newer.

When you install Red Hat Advanced Cluster Security for Kubernetes, there is no default network baseline. As Red Hat Advanced Cluster Security for Kubernetes discovers network flows, it creates a baseline and then it adds all discovered network flows to it.

- When Red Hat Advanced Cluster Security for Kubernetes discovers new network activity, it adds that network flow to the network baseline.

- Network flows do not show up as anomalous flows and do not trigger any violations.

After the discovery phase:

- Red Hat Advanced Cluster Security for Kubernetes stops adding network flows to the network baselines.

- New network flows (which are not in the network baseline) show up as anomalous flows but they do not trigger any violations.

8.5.1. Viewing network baselines

You can view network baselines from the network graph view.

Procedure

1. On the **Network Graph** view, select one or more namespaces.

2. Select a deployment.

3. The Network Flows details panel show both anomalous and baseline flows. From this panel, you can:

   - mark network flows from the baseline as anomalous by selecting **Mark as Anomalous**, or

   - add network flows to baseline from the anomalous flows by selecting **Add to Baseline**.

8.5.2. Enabling alerts on baseline violations

You can configure Red Hat Advanced Cluster Security for Kubernetes to detect anomalous network flows and trigger violations.

> **NOTE**
>
> You need Red Hat Advanced Cluster Security for Kubernetes version 3.0.56 or newer to enable alerts on baseline violations.

Procedure

1. On the **Network Graph** view, select a deployment.

2. In the network flow details panel, select **Baseline Settings**.

3. Turn on the **Alert on baseline violations** toggle.

   - Once you turn on the **Alert on baseline violations** toggle anomalous network flows trigger violations.

   - You can turn off the **Alert on baseline violations** toggle to stop receiving violations for anomalous network flows.

CHAPTER 9. REVIEWING CLUSTER CONFIGURATION

Learn how to use the **Configuration Management** view and understand the correlation between various entities in your cluster to manage your cluster configuration efficiently.

Every OpenShift Container Platform cluster includes many different entities distributed throughout the cluster, which makes it more challenging to understand and act on the available information.

Red Hat Advanced Cluster Security for Kubernetes (RHACS) provides efficient configuration management that combines all these distributed entities on a single page. It brings together information about all your clusters, namespaces, nodes, deployments, images, secrets, users, groups, service accounts, and roles in a single **Configuration Management** view, helping you visualize different entities and the connections between them.

9.1. USING THE CONFIGURATION MANAGEMENT VIEW

To open the **Configuration Management** view, select **Configuration Management** from the navigation menu. Similar to the **Dashboard**, it displays some useful widgets.

These widgets are interactive and show the following information:

- Security policy violations by severity

- The state of CIS (Center for Information Security) Docker and Kubernetes benchmark controls

- Users with administrator rights in the most clusters

- Secrets used most widely in your clusters

The header in the **Configuration Management** view shows you the number of policies and CIS controls in your cluster.

> **NOTE**
>
> Only policies in the Deploy life cycle phase are included in the policy count and policy list view.

The header includes drop-down menus that allow you to switch between entities. For example, you can:

- Click **Policies** to view all policies and their severity, or select   **CIS Controls** to view detailed information about all controls.

- Click **Application and Infrastructure** and select clusters, namespaces, nodes, deployments, images, and secrets to view detailed information.

- Click **RBAC Visibility and Configuration** and select users and groups, service accounts, and roles to view detailed information.

9.2. IDENTIFYING MISCONFIGURATIONS IN KUBERNETES ROLES

You can use the **Configuration Management** view to identify potential misconfigurations, such as users, groups, or service accounts granted the **cluster-admin** role, or roles that are not granted to anyone.

9.2.1. Finding Kubernetes roles are their assignment

Use the **Configuration Management** view to get information about which Kubernetes roles are assigned to specific users and groups.

Procedure

1. Navigate to the RHACS portal and click **Configuration Management** from the left-hand navigation menu.

2. Select **RBAC Visibility and Configuration → Users and Groups** from the header in the **Configuration Management** view. The **Users and Groups** view displays a list of Kubernetes users and groups, their assigned roles, and whether the **cluster-admin** role is enabled for each of them.

3. Select a user or group to view more details about the associated cluster and namespace permissions.

9.2.2. Finding service accounts and their permissions

Use the **Configuration Management** view to find out where service accounts are in use and their permissions.

Procedure

1. Navigate to the RHACS portal and click **Configuration Management** from the left-hand navigation menu.

2. Select **RBAC Visibility and Configuration → Service Accounts** from the header in the **Configuration Management** view. The **Service Accounts** view displays a list of Kubernetes service accounts across your clusters, their assigned roles, whether the **cluster-admin** role is enabled, and which deployments use them.

3. Select a row or an underlined link to view more details, including which cluster and namespace permissions are granted to the selected service account.

9.2.3. Finding unused Kubernetes roles

Use the **Configuration Management** view to get more information about your Kubernetes roles and find unused roles.

Procedure

1. Navigate to the RHACS portal and click **Configuration Management** from the left-hand navigation menu.

2. Select **RBAC Visibility and Configuration → Roles** from the header in the **Configuration Management** view. The **Roles** view displays a list of Kubernetes roles across your clusters, the permissions they grant, and where they are used.

3. Select a row or an underlined link to view more details about the role.

4. To find roles not granted to any users, groups, or service accounts, select the **Users & Groups** column header. Then select the **Service Account** column header while holding the **Shift** key. The list shows the roles that are not granted to any users, groups, or service accounts.

9.3. VIEWING KUBERNETES SECRETS

View Kubernetes secrets in use in your environment and identify deployments using those secrets.

Procedure

1. Navigate to the RHACS portal and click **Configuration Management** from the left-hand navigation menu.

2. On the **Secrets Most Used Across Deployments** widget, select **View All**. The **Secrets** view displays a list of Kubernetes secrets.

3. Select a row to view more details.

Use the available information to identify if the secrets are in use in deployments where they are not needed.

9.4. FINDING POLICY VIOLATIONS

The **Policy Violations by Severity** widget in the **Configuration Management** view displays policy violations in a sunburst chart. Each level of the chart is represented by one ring or circle.

- The innermost circle represents the total number of violations.

- The next ring represents the **Low**, **Medium**, **High**, and **Critical** policy categories.

- The outermost ring represents individual policies in a particular category.

The **Configuration Management** view only shows the information about policies that have the **Lifecycle Stage** set to **Deploy**. It does not include policies that address runtime behavior or those configured for assessment in the **Build** stage.

Procedure

1. Navigate to the RHACS portal and click **Configuration Management** from the left-hand navigation menu.

2. On the **Policy Violations by Severity** widget, move your mouse over the sunburst chart to view details about policy violations.

3. Select *n* **rated as high**, where *n* is a number, to view detailed information about high-priority policy violations. The **Policies** view displays a list of policy violations filtered on the selected category.

4. Select a row to view more details, including policy description, remediation, deployments with violations, and more. The details are visible in a panel.

5. The **Policy Findings** section in the information panel lists deployments where these violations occurred.

6. Select a deployment under the **Policy Findings** section to view related details including Kubernetes labels, annotations, and service account.

You can use the detailed information to plan a remediation for violations.

9.5. FINDING FAILING CIS CONTROLS

Similar to the **Policy Violations** sunburst chart in the **Configuration Management** view, the **CIS controls** widget provides information about failing Center for Information Security (CIS) controls.

Each level of the chart is represented by one ring or circle.

- The innermost circle represents the percentage of failing controls.

- The next ring represents the control categories.

- The outermost ring represents individual controls in a particular category.

Procedure

1. Select **CIS Docker v1.2.0** from the header of the **CIS controls** widget. Use this to switch between CIS Docker and Kubernetes controls.

2. Hover over the sunburst chart to view details about failing controls.

3. Select *n* **controls failing**, where *n* is a number, to view detailed information about failing controls. The **Controls** view displays a list of failing controls filtered based on the compliance state.

4. Select a row to view more details, including control descriptions and nodes where the controls are failing.

5. The **Control Findings** section in the information panel lists nodes where the controls are failing. Select a row to view more details, including Kubernetes labels, annotations, and other metadata.

You can use the detailed information to focus on a subset of nodes, industry standards, or failing controls. You can also assess, check, and report on the compliance status of your containerized infrastructure.

CHAPTER 10. EXAMINING IMAGES FOR VULNERABILITIES

With Red Hat Advanced Cluster Security for Kubernetes you can analyze images for vulnerabilities. Scanner analyzes all image layers to check for known vulnerabilities by comparing them with the Common Vulnerabilities and Exposures (CVEs) list.

When Scanner finds any vulnerabilities, it:

- Shows them in the **Vulnerability Management** view for detailed analysis.

- Ranks vulnerabilities according to risk and highlights them in the RHACS portal for risk assessment.

- Checks them against enabled security policies.

Scanner inspects the images and identifies the installed components based on the files in the images. It may fail to identify installed components or vulnerabilities if the final images are modified to remove the following files:

Components	Files
Package managers	**/etc/alpine-release** **/etc/apt/sources.list** **/etc/lsb-release** **/etc/os-release** or **/usr/lib/os-release** **/etc/oracle-release**, **/etc/centos-release**, **/etc/redhat-release**, or **/etc/system-release** - Other similar system files.
Language-level dependencies	**package.json** for JavaScript. **dist-info** or **egg-info** for Python. **MANIFEST.MF** in Java Archive (JAR) for Java.
Application-level dependencies	**dotnet/shared/Microsoft.AspNetCore.App/** **dotnet/shared/Microsoft.NETCore.App/**

10.1. SCANNING IMAGES

Central submits image scanning requests to Scanner. Upon receiving these requests, Scanner pulls the image layers from the relevant registry, checks the images, and identifies installed packages in each

layer. Then it compares the identified packages and programming language-specific dependencies with the vulnerability lists and sends information back to Central.

You can also integrate Red Hat Advanced Cluster Security for Kubernetes with another vulnerability scanner.

Scanner identifies the vulnerabilities in the:

- base image operating system
- packages that are installed by the package managers
- programming language specific dependencies
- programming runtimes and frameworks

Understanding and addressing common Scanner warning messages

When scanning images with Red Hat Advanced Cluster Security for Kubernetes (RHACS), you might see the **CVE DATA MAY BE INACCURATE** warning message. Scanner displays this message when it cannot retrieve complete information about the operating system or other packages in the image.

The following table shows some common Scanner warning messages:

Table 10.1. Warning messages

Message	Description
Unable to retrieve the OS CVE data, only Language CVE data is available	Indicates that Scanner does not officially support the base operating system of the image; therefore, it cannot retrieve CVE data for the operating system-level packages.
Stale OS CVE data	Indicates that the base operating system of the image has reached end-of-life, which means the vulnerability data is outdated. For example, Debian 8 and 9. For more information about the files needed to identify the components in the images, see Examining images for vulnerabilities.
Failed to get the base OS information	Indicates that Scanner scanned the image, but was unable to determine the base operating system used for the image.
Failed to retrieve metadata from the registry	Indicates that the target registry is unreachable on the network. The cause could be a firewall blocking **docker.io**, or an authentication issue preventing access. To analyze the root cause, create a special registry integration for private registries or repositories to get the pod logs for RHACS Central. For instructions on how to do this, see Integrating with image registries

Message	Description
Image out of scope for Red Hat Vulnerability Scanner Certification	Indicates that Scanner scanned the image, but the image is old and does not fall within the scope of Red Hat Scanner Certification. For more information, see Partner Guide for Red Hat Vulnerability Scanner Certification. **IMPORTANT** If you are using a Red Hat container image, consider using a base image newer than June 2020.

Supported package formats

Scanner can check for vulnerabilities in images that use the following package formats:

- yum

- microdnf

- apt

- apk

- dpkg

- RPM

Supported programming languages

Scanner can check for vulnerabilities in dependencies for the following programming languages:

- Java

- JavaScript

- Python

- Ruby

Supported runtimes and frameworks

Beginning from Red Hat Advanced Cluster Security for Kubernetes 3.0.50 (Scanner version 2.5.0), Scanner identifies vulnerabilities in the following developer platforms:

- .NET Core

- ASP.NET Core

Supported operating systems

The supported platforms listed in this section are the distributions in which Scanner identifies vulnerabilities, and it is different from the supported platforms on which you can install Red Hat Advanced Cluster Security for Kubernetes.

Scanner identifies vulnerabilities in images that contain the following Linux distributions:

Distribution	Version
Alpine Linux	alpine:v3.2, alpine:v3.3, alpine:v3.4, alpine:v3.5, alpine:v3.6, alpine:v3.7, alpine:v3.8, alpine:v3.9, alpine:v3.10, alpine:v3.11, alpine:v3.12, alpine:v3.13, alpine:v3.14, alpine:v3.15, alpine:v3.16, alpine:edge
Amazon Linux	amzn:2018.03, amzn:2
CentOS	centos:6, centos:7, centos:8
Debian	debian:9, debian:10, debian:11, debian:unstable
Red Hat Enterprise Linux (RHEL)	rhel:6, rhel:7, rhel:8, rhel:9
Ubuntu	ubuntu:14.04, ubuntu:16.04, ubuntu:18.04, ubuntu:20.04, ubuntu:21.10, ubuntu:22.04

NOTE

- Scanner does not support the Fedora operating system because Fedora does not maintain a vulnerability database. However, Scanner still detects language-specific vulnerabilities in Fedora-based images.

- Scanner also identifies vulnerabilities in the following images. However, the vulnerability sources are not updated anymore by their vendor:

Distribution	Version
Debian	debian:8
Ubuntu	ubuntu:12.04, ubuntu:12.10, ubuntu:13.04, ubuntu:14.10, ubuntu:15.04, ubuntu::15.10, ubuntu::16.10, ubuntu:17.04, ubuntu:17.10, ubuntu:18.10, ubuntu:19.04, ubuntu:19.10, ubuntu:20.10, ubuntu:21.04

10.2. PERIODIC SCANNING OF IMAGES

Red Hat Advanced Cluster Security for Kubernetes periodically scans all active images and updates the image scan results to reflect the latest vulnerability definitions. Active images are the images you have deployed in your environment.

> **NOTE**
>
> From Red Hat Advanced Cluster Security for Kubernetes 3.0.57, you can enable automatic scanning of inactive images by configuring the **Watch** setting for images.

Central fetches the image scan results for all active images from Scanner or other integrated image scanners that you use and updates the results every 4 hours.

You can also use the **roxctl** CLI to check the image scan results on demand.

10.3. SCANNING INACTIVE IMAGES

Red Hat Advanced Cluster Security for Kubernetes (RHACS) scans all active (deployed) images every 4 hours and updates the image scan results to reflect the latest vulnerability definitions.

You can also configure RHACS to scan inactive (not deployed) images automatically.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management > Dashboard**

2. On the **Dashboard** view header, select **IMAGES**.

3. Click **MANAGE WATCHES** to manage the scanning of watched images.

4. In the **MANAGE WATCHED IMAGES** dialog, enter the inactive image's name for which you want to enable scanning.
   Verify that you enter the name of the image and not the image **id**. Image name is the fully-qualified image name, beginning with the registry and ending with the tag. For example, **docker.io/vulnerables/cve-2017-7494:latest**.

5. Select **ADD IMAGE**. RHACS then scans the image and shows the error or success message.

6. (Optional) Click **REMOVE WATCH** to remove an image from the watchlist.

> **IMPORTANT**
>
> On the RHACS portal, click **Platform Configuration > System Configuration** to view the data retention configuration.
>
> All the data related to the image removed from the watchlist continues to appear in the RHACS portal for the number of days mentioned on the **System Configuration** page and is only removed after that period is over.

7. Select **RETURN TO IMAGE LIST** to view the **IMAGES** page.

10.4. FETCHING VULNERABILITY DEFINITIONS

In online mode, Central fetches the vulnerability definitions every 5 minutes from a single feed. This feed combines vulnerability definitions from upstream sources that include multiple Linux distributions and the National Vulnerability Database, and it refreshes every hour.

- The address of the feed is **https://definitions.stackrox.io**.

- You can change the default query frequency for Central by setting the **ROX_SCANNER_VULN_UPDATE_INTERVAL** environment variable:

  ```
  $ oc -n stackrox set env deploy/central ROX_SCANNER_VULN_UPDATE_INTERVAL=
  <value> 1
  ```

  1   If you use Kubernetes, enter **kubectl** instead of **oc**.
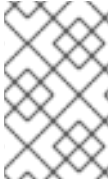
> **NOTE**
>
> Scanner's configuration map still has an **updater.interval** parameter for configuring Scanner's updating frequency, but it no longer includes the **fetchFromCentral** parameter.

10.5. UNDERSTANDING VULNERABILITY SCORES

In the RHACS portal shows a single Common Vulnerability Scoring System (CVSS) base score for each vulnerability. Red Hat Advanced Cluster Security for Kubernetes shows the CVSS score based on the following criteria:

- If a CVSS v3 score is available, Red Hat Advanced Cluster Security for Kubernetes shows the score and lists **v3** along with it. For example, **6.5 (v3)**.
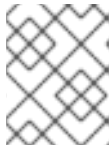
  > **NOTE**
  >
  > CVSS v3 scores are only available if you are using Scanner version 1.3.5 and newer.

- If a CVSS v3 score is not available, Red Hat Advanced Cluster Security for Kubernetes shows only the CVSS v2 score. For example, **6.5**.

You can use the API to get the CVSS scores. If CVSS v3 information is available for a Common Vulnerabilities and Exposures (CVE), the response includes both CVSS v3 and CVSS v2 information.

For some CVEs, the Red Hat Security Advisory (RHSA) CVSS score may differ from the CVSS score visible in the RHACS portal. This difference is because one RHSA can contain multiple CVEs, and Red Hat sometimes assigns a different score based on how a vulnerability affects other Red Hat products.

In such cases, Red Hat Advanced Cluster Security for Kubernetes:

- Finds the highest–scoring CVE from the National Vulnerability Database (NVD) and shows its score as the CVSS score for the RHSA.

- Breaks out each CVE in the RHSA as a separate vulnerability with its original CVSS score (from the NVD), so that you can view each one and create policies for specific CVEs.

10.5.1. Additional resources

- Scanning inactive images

- Getting started with the roxctl CLI

10.6. VIEWING IMAGES IN YOUR ENVIRONMENT

With Red Hat Advanced Cluster Security for Kubernetes you can view the details for all container images in your clusters.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the left-hand navigation menu.

2. To view details for all the images in your cluster, select **Images** on the **Vulnerability Management** view header.

10.7. VIEWING THE DOCKERFILE FOR AN IMAGE

Use the **Vulnerability Management** view to find the root cause of vulnerabilities in an image. You can view the Dockerfile and find exactly which command in the Dockerfile introduced the vulnerabilities and all components that are associated with that single command.

The Dockerfile section shows information about:

- All the layers in the Dockerfile

- The instructions and their value for each layer

- The components included in each layer

- The number of CVEs in components for each layer

When there are components introduced by a specific layer, you can select the expand icon to see a summary of its components. If there are any CVEs in those components, you can select the expand icon for an individual component to get more details about the CVEs affecting that component.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select an image from either the **Top Riskiest Images** widget or click the **Images** button at the top of the Dashboard and select an image.

3. In the **Image** details view, next to **Dockerfile**, select the expand icon to see a summary of instructions, values, creation date, and components.

4. Select the expand icon for an individual component to view more information.

10.8. IDENTIFYING THE CONTAINER IMAGE LAYER THAT INTRODUCES VULNERABILITIES

Use the **Vulnerability Management** view to identify vulnerable components and the image layer they appear in.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select an image from either the **Top Riskiest Images** widget or click the **Images** button at the top of the Dashboard and select an image.

3. In the **Image** details view, next to **Dockerfile**, select the expand icon to see a summary of image components.

4. Select the expand icon for specific components to get more details about the CVEs affecting the selected component.

10.9. IDENTIFYING DOCKERFILE LINES IN IMAGES THAT INTRODUCED COMPONENTS WITH CVES

You can identify specific Dockerfile lines in an image that introduced components with CVEs.

Procedure

To view a problematic line:

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select an image from either the **Top Riskiest Images** widget or click the **Images** button at the top of the Dashboard and select an image.

3. In the **Image** details view, under **Image Findings**, CVEs are listed in the **Observed CVEs**, **Deferred CVEs**, and **False positive CVEs** tabs.

4. Locate the CVE you want to examine further. In the **Affected Components** column, click on the **<number> Components** link to view a list of components affected by the CVE. You can perform the following actions in this window:

   - Select the expand icon next to a specific component to view the Dockerfile line in the image that introduced the CVE. To address the CVE, you need to change this line in the Dockerfile; for example, you can upgrade the component.

   - Click the name of the component to go to the **Component Summary** page and view more information about the component.

10.10. IDENTIFYING THE OPERATING SYSTEM OF THE BASE IMAGE

Use the **Vulnerability Management** view to identify the operating system of the base image.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. From the **Vulnerability Management** view header, select **Images**.

3. View the base operating system (OS) and OS version for all images under the **Image OS** column.

4. Select an image to view its details. The base operating system is also available under the **Image Summary → Details and Metadata** section.

NOTE

Red Hat Advanced Cluster Security for Kubernetes lists the **Image OS** as **unknown** when either:

- The operating system information is not available, or

- If the image scanner in use does not provide this information.

Docker Trusted Registry, Google Container Registry, and Anchore do not provide this information.

10.11. DISABLING LANGUAGE-SPECIFIC VULNERABILITY SCANNING

Scanner identifies the vulnerabilities in the programming language-specific dependencies by default. You can disable the language-specific dependency scanning.

Procedure

- To disable language-specific vulnerability scanning, run the following command:

```
$ oc -n stackrox set env deploy/scanner \ 1
  ROX_LANGUAGE_VULNS=false 2
```

1 If you use Kubernetes, enter **kubectl** instead of **oc**.

2 If you are using Red Hat Advanced Cluster Security for Kubernetes version 3.0.47 or older, replace the environment variable name **ROX_LANGUAGE_VULNS** with **LANGUAGE_VULNS**.

10.12. ADDITIONAL RESOURCES

- For more information about Common Vulnerabilities and Exposures (CVEs), see the Red Hat CVE Database.

CHAPTER 11. VERIFYING IMAGE SIGNATURES

You can use Red Hat Advanced Cluster Security for Kubernetes (RHACS) to ensure the integrity of the container images in your clusters by verifying image signatures against pre-configured keys.

You can create policies to block unsigned images and images that do not have a verified signature. You can also enforce the policy by using the RHACS admission controller to stop unauthorized deployment creation.

> **NOTE**
>
> - RHACS 3.70 only supports Cosign signatures and Cosign public key signature verification. For more information about Cosign, see Cosign overview.
>
> - You must configure signature integration with at least 1 Cosign public key for signature verification.
>
> - For all deployed and watched images:
>
>   - RHACS fetches and verifies the signatures every 4 hours.
>
>   - RHACS verifies the signatures whenever you change or update your signature integration public keys.

11.1. CONFIGURING SIGNATURE INTEGRATION

Before performing image signature verification, you must first add your Cosign public keys in RHACS.

Prerequisites

- You must already have a PEM-encoded Cosign public key. For more information about Cosign, see Cosign overview.

Procedure

1. On the RHACS portal, select **Platform Configuration → Integrations**.

2. Scroll down to the **Signature Integrations** section and click **Signature**.

3. Click **New integration**.

4. Enter a name for the **Integration name**.

5. Click **Cosign → Add a new public key**.

6. Enter the **Public key** name.

7. For the **Public key value** field, enter the PEM-encoded public key.

8. (Optional) You can add more than one key by clicking **Add a new public key** and entering the details.

9. Click **Save**.

11.2. USING SIGNATURE VERIFICATION IN A POLICY

When creating custom security policies, you can use the **Trusted image signers** policy criteria to verify image signatures.

Prerequisites

- You must have already configured a signature integration with at least 1 Cosign public key.

Procedure

1. When creating or editing a policy, drag the **Not verified by trusted image signers** policy criteria in the policy field drop area for the **Policy criteria** section.

2. Click **Select**.

3. Select the trusted image signers from the list and click **Save**.

Additional resources

- Creating a security policy from the system policies view

- Policy criteria

11.3. ENFORCING SIGNATURE VERIFICATION

To prevent the users from using unsigned images, you can enforce signature verification by using the RHACS admission controller. You must first enable the **Contact Image Scanners** feature in your cluster configuration settings. Then, while creating a security policy to enforce signature verification, you can use the **Inform and enforce** option.

For more information, see Enabling admission controller enforcement.

Additional resources

- Creating a security policy from the system policies view

CHAPTER 12. MANAGING VULNERABILITIES

Security vulnerabilities in your environment may be exploited by an attacker to perform unauthorized actions such as denial of service, remote code execution, or unauthorized access to sensitive data. Therefore, the management of vulnerabilities is a foundational step towards a successful Kubernetes security program.

12.1. VULNERABILITY MANAGEMENT PROCESS

Vulnerability management is a continuous process to identify and remediate vulnerabilities. Red Hat Advanced Cluster Security for Kubernetes helps you to facilitate a vulnerability management process.

A successful vulnerability management program often includes the following critical tasks:

- Performing asset assessment

- Prioritizing the vulnerabilities

- Assessing the exposure

- Taking action

- Continuously reassessing assets

Red Hat Advanced Cluster Security for Kubernetes helps organizations to perform continuous assessments on their OpenShift Container Platform and Kubernetes clusters. It provides organizations with the contextual information they need to prioritize and act on vulnerabilities in their environment more effectively.

12.1.1. Performing asset assessment

Performing an assessment of an organization's assets involve the following actions:

- Identifying the assets in your environment.

- Scanning these assets to identify known vulnerabilities.

- Reporting on the vulnerabilities in your environment to impacted stakeholders.

When you install Red Hat Advanced Cluster Security for Kubernetes on your Kubernetes or OpenShift Container Platform cluster, it first aggregates the assets running inside of your cluster to help you identify those assets.Red Hat Advanced Cluster Security for Kubernetes allows organizations to perform continuous assessments on their OpenShift Container Platform and Kubernetes clusters. It provides organizations with the contextual information to prioritize and act on vulnerabilities in their environment more effectively.

Important assets that should be monitored by the organization's vulnerability management process using Red Hat Advanced Cluster Security for Kubernetes include:

- **Components**: Components are software packages that may be used as part of an image or run on a node. Components are the lowest level where vulnerabilities are present. Therefore, organizations must upgrade, modify or remove software components in some way to remediate vulnerabilities.

- **Image**: A collection of software components and code that create an environment to run an executable portion of code. Images are where you upgrade components to fix vulnerabilities.

- **Nodes**: A server used to manage and run applications using OpenShift or Kubernetes and the components that make up the OpenShift Container Platform or Kubernetes service.

Red Hat Advanced Cluster Security for Kubernetes groups these assets into the following structures:

- **Deployment**: A definition of an application in Kubernetes that may run pods with containers based on one or many images.

- **Namespace**: A grouping of resources such as Deployments that support and isolate an application.

- **Cluster**: A group of nodes used to run applications using OpenShift or Kubernetes.

Red Hat Advanced Cluster Security for Kubernetes scans the assets for known vulnerabilities and uses the Common Vulnerabilities and Exposures (CVE) data to assess the impact of a known vulnerability.

12.1.1.1. Viewing application vulnerabilities

You can view application vulnerabilities in Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Application & Infrastructure → Namespaces** or **Deployments**.

3. From the list, search for and select the **Namespace** or **Deployment** you want to review.

4. To get more information about the application, select an entity from **Related entities** on the right.

12.1.1.2. Viewing image vulnerabilities

You can view image vulnerabilities in Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Images**.

3. From the list of images, select the image you want to investigate. You can also filter the list by performing one of the following steps:

   a. Enter **Image** in the search bar and then select the **Image** attribute.

   b. Enter the image name in the search bar.

4. In the image details view, review the listed CVEs and prioritize taking action to address the impacted components.

5. Select **Components** from **Related entities** on the right to get more information about all the components that are impacted by the selected image. Or select **Components** from the **Affected components** column under the **Image findings** section for a list of components affected by specific CVEs.

Additional resources

- Using local page filtering

12.1.1.3. Viewing infrastructure vulnerabilities

You can view vulnerabilities in nodes by using Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Application & Infrastructure → Cluster**.

3. From the list of clusters, select the cluster you want to investigate.

4. Review the clusters vulnerabilities and prioritize taking action on the impacted nodes on the cluster.

12.1.1.4. Viewing node vulnerabilities

You can view vulnerabilities in specific nodes by using Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Nodes**.

3. From the list of nodes, select the node you want to investigate.

4. Review vulnerabilities for the selected node and prioritize taking action.

5. To get more information about the affected components in a node, select **Components** from **Related entities** on the right.

12.1.2. Prioritizing the vulnerabilities

Answer the following questions to prioritize the vulnerabilities in your environment for action and investigation:

- How important is an affected asset for your organization?

- How severe does a vulnerability need to be for investigation?

- Can the vulnerability be fixed by a patch for the affected software component?

- Does the existence of the vulnerability violate any of your organization's security policies?

The answers to these questions help security and development teams decide if they want to gauge the exposure of a vulnerability.

Red Hat Advanced Cluster Security for Kubernetes provides you the means to facilitate the prioritization of the vulnerabilities in your applications and components.

12.1.3. Assessing the exposure

To assess your exposure to a vulnerability, answer the following questions:

- Is your application impacted by a vulnerability?

- Is the vulnerability mitigated by some other factor?

- Are there any known threats that could lead to the exploitation of this vulnerability?

- Are you using the software package which has the vulnerability?

- Is spending time on a specific vulnerability and the software package worth it?

Take some of the following actions based on your assessment:

- Consider marking the vulnerability as a false positive if you determine that there is no exposure or that the vulnerability does not apply in your environment.

- Consider if you would prefer to remediate, mitigate or accept the risk if you are exposed.

- Consider if you want to remove or change the software package to reduce your attack surface.

12.1.4. Taking action

Once you have decided to take action on a vulnerability, you can take one of the following actions:

- Remediate the vulnerability

- Mitigate and accept the risk

- Accept the risk

- Mark the vulnerability as a false positive

You can remediate vulnerabilities by performing one of the following actions:

- Remove a software package

- Update a software package to a non-vulnerable version.

Additional resources

- Reviewing a false positive or deferred CVE

12.1.4.1. Finding a new component version

The following procedure finds a new component version to upgrade to.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Images**.

3. From the list of images, select the image you already assessed.

4. Under the **Image findings** section, select the CVE.

5. Select the Affected components of the CVE you want to take action on.

6. Review the version of the component that the CVE is fixed in and update your image.

12.1.5. Accepting risks

Follow the instructions in this section to accept the risks in Red Hat Advanced Cluster Security for Kubernetes.

Prerequisites

- You must have **write** permission for the **VulnerabilityManagementRequests** resource.

To accept risk with or without mitigation:

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Images**.

3. From the list of images, select the image you already assessed.

4. Find the row which lists the CVE you would like to take action on.

5. Click ⋮ on the right for the CVE you identified and click **Defer CVE**.

6. Select the date and time till you want to defer the CVE.

7. Select if you want to defer the CVE for the selected image tag or all tags for this image.

8. Enter the reason for the deferral.

9. Click **Request approval**. Select the blue information icon on the right of the CVE and copy the approval link to share with your organization's deferral approver.

12.1.5.1. Marking vulnerabilities as false positive

The following procedure marks a vulnerability as a false positive.

Prerequisites

- You must have the **write** permission for the **VulnerabilityManagementRequests** resource.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Dashboard**.

2. On the **Dashboard** view header, select **Images**.

3. From the list of images, select the image you already assessed.

4. Find the row which lists the CVE you would like to take action on.

5. Click the ⋮ on the right for the CVE you identified and click **Defer CVE**.

6. Select the date and time you want to defer the CVE.

7. Select if you want to defer the CVE for the selected image tag or all tags for this image.

8. Enter the reason for the deferral.

9. Click **Request approval**.

10. Select the blue information icon on the right of the CVE and copy the approval link to share with your organization's deferral approver.

12.1.5.2. Reviewing a false positive or deferred CVE

Use the following procedure to review a false positive or deferred CVE.

Prerequisites

- You must have the **write** permission for the **VulnerabilityManagementApprovals** resource.

You can review a false positive or defered CVE:

Procedure

1. Open the approval link in your browser or in the RHACS portal.

2. Navigate to **Vulnerability Management** → **Risk Acceptance** and search for the CVE.

3. Review the vulnerabilities scope and action to decide if you would like to approve it.

4. Click on the ⋮ at the far right of the CVE and approve or deny the request for approval.

12.1.6. Reporting vulnerabilities to teams

As organizations must constantly reassess and report on their vulnerabilities, some organizations find it helpful to have scheduled communications to key stakeholders to help in the vulnerability management process.

You can use Red Hat Advanced Cluster Security for Kubernetes to schedule these reoccurring communications through e-mail. These communications should be scoped to the most relevant information that the key stakeholders need.

For sending these communications, you must consider the following questions:

- What schedule would have the most impact when communicating with the stakeholders?

- Who is the audience?

- Should you only send specific severity vulnerabilities in your report?

- Should you only send fixable vulnerabilities in your report?

12.1.6.1. Scheduling vulnerability management reports

The following procedure creates a scheduled vulnerability report.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Reporting**.

2. Click **Create report**.

3. Enter a name for your report in the **Report name** field.

4. Select a weekly or monthly cadence for your report under **Repeat report…**

5. Enter **Description** for the report.

6. Select the scope for the report by selecting if you want to report fixable vulnerabilities, vulnerabilities of a specific severity, or vulnerabilities that only appeared since the last scheduled report.

7. For **Configure resource scope**, select the scope of resources the vulnerabilities apply to.

8. Select or create an e-mail notifier to send your report by e-mail and configure your distribution list under **Notification and distribution**.

9. Select **Create** to schedule the report.

12.1.6.2. Sending a vulnerability report

The following procedure sends a vulnerability report.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Reporting**.

2. From the list of reports, select the report.

3. Select the ⋮ on the right of the report and click **Run report now**.

12.1.6.3. Editing a vulnerability report

The following procedure edits a vulnerability report.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Reporting**.

2. From the list of reports, select the report.

3. Select the ⋮ on the right of the report and click **Edit**.

4. Modify the report as required.

5. Click **Save**.

12.1.6.4. Deleting a vulnerability report

The following procedure deletes a vulnerability report.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management → Reporting**.

2. From the list of reports, select the report.

3. Select the ⋮ on the right of the report and click **Delete report**.

12.2. COMMON TASKS

This section lists some common tasks you can perform from the **Vulnerability Management → Dashboard** view.

12.2.1. Finding critical CVEs impacting your infrastructure

Use the **Vulnerability Management** view for identifying CVEs that are impacting your platform the most.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select CVEs on the **Vulnerability Management** view header.

3. In the **CVEs** view, select the **Env Impact** column header to arrange the CVEs in descending order (highest first) based on the environment impact.

12.2.2. Finding the most vulnerable image components

Use the **Vulnerability Management** view for identifying highly vulnerable image components.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. From the **Vulnerability Management** view header, select **Application & Infrastructure → Components**.

3. In the **Components** view, select the **CVEs** column header to arrange the components in descending order (highest first) based on the CVEs count.

12.2.3. Identifying the container image layer that introduces vulnerabilities

Use the **Vulnerability Management** view to identify vulnerable components and the image layer they appear in.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select an image from either the **Top Riskiest Images** widget or click the **Images** button at the top of the Dashboard and select an image.

3. In the **Image** details view, next to **Dockerfile**, select the expand icon to see a summary of image components.

4. Select the expand icon for specific components to get more details about the CVEs affecting the selected component.

12.2.4. Viewing details only for fixable CVEs

Use the **Vulnerability Management** view to filter and show only the fixable CVEs.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. From the **Vulnerability Management** view header, select **Filter CVEs → Fixable**.

12.2.5. Identifying the operating system of the base image

Use the **Vulnerability Management** view to identify the operating system of the base image.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. From the **Vulnerability Management** view header, select **Images**.

3. View the base operating system (OS) and OS version for all images under the **Image OS** column.

4. Select an image to view its details. The base operating system is also available under the **Image Summary → Details and Metadata** section.

> **NOTE**
>
> Red Hat Advanced Cluster Security for Kubernetes lists the **Image OS** as **unknown** when either:
>
> - The operating system information is not available, or
>
> - If the image scanner in use does not provide this information.
>
> Docker Trusted Registry, Google Container Registry, and Anchore do not provide this information.

12.2.6. Identifying top risky objects

Use the **Vulnerability Management** view for identifying the top risky objects in your environment. The **Top Risky** widget displays information about the top risky images, deployments, clusters, and namespaces in your environment. The risk is determined based on the number of vulnerabilities and their CVSS scores.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select the **Top Risky** widget header to choose between riskiest images, deployments, clusters, and namespaces.
   The small circles on the chart represent the chosen object (image, deployment, cluster, namespace). Hover over the circles to see an overview of the object they represent. And select a circle to view detailed information about the selected object, its related entities, and the connections between them.

   For example, if you are viewing **Top Risky Deployments by CVE Count and CVSS score**, each circle on the chart represents a deployment.

   - When you hover over a deployment, you see an overview of the deployment, which includes deployment name, name of the cluster and namespace, severity, risk priority, CVSS, and CVE count (including fixable).

   - When you select a deployment, the **Deployment** view opens for the selected deployment. The **Deployment** view shows in-depth details of the deployment and includes information about policy violations, common vulnerabilities, CVEs, and riskiest images for that deployment.

3. Select **View All** on the widget header to view all objects of the chosen type. For example, if you chose **Top Risky Deployments by CVE Count and CVSS score**, you can select **View All** to view detailed information about all deployments in your infrastructure.

12.2.7. Identifying top riskiest images and components

Similar to the **Top Risky**, the **Top Riskiest** widget lists the names of the top riskiest images and components. This widget also includes the total number of CVEs and the number of fixable CVEs in the listed images.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select the **Top Riskiest Images** widget header to choose between the riskiest images and components. If you are viewing **Top Riskiest Images**:

   - When you hover over an image in the list, you see an overview of the image, which includes image name, scan time, and the number of CVEs along with severity (critical, high, medium, and low).

   - When you select an image, the **Image** view opens for the selected image. The **Image** view shows in-depth details of the image and includes information about CVEs by CVSS score, top riskiest components, fixable CVEs, and Dockerfile for the image.

3. Select **View All** on the widget header to view all objects of the chosen type. For example, if you chose **Top Riskiest Components**, you can select **View All** to view detailed information about all components in your infrastructure.

12.2.8. Viewing the Dockerfile for an image

Use the **Vulnerability Management** view to find the root cause of vulnerabilities in an image. You can view the Dockerfile and find exactly which command in the Dockerfile introduced the vulnerabilities and all components that are associated with that single command.

The Dockerfile section shows information about:

- All the layers in the Dockerfile

- The instructions and their value for each layer

- The components included in each layer

- The number of CVEs in components for each layer

When there are components introduced by a specific layer, you can select the expand icon to see a summary of its components. If there are any CVEs in those components, you can select the expand icon for an individual component to get more details about the CVEs affecting that component.

Procedure

1. Navigate to the RHACS portal and click **Vulnerability Management** from the navigation menu.

2. Select an image from either the **Top Riskiest Images** widget or click the **Images** button at the top of the Dashboard and select an image.

3. In the **Image** details view, next to **Dockerfile**, select the expand icon to see a summary of instructions, values, creation date, and components.

4. Select the expand icon for an individual component to view more information.

12.2.9. Disabling identifying vulnerabilities in nodes

Identifying vulnerabilities in nodes is enabled by default. You can disable it from the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Integrations**.

2. Under **Image Integrations**, select **StackRox Scanner**.

3. From the list of scanners, select **StackRox Scanner** to view its details.

4. Remove the **Node Scanner** option from **Types**.

5. Select **Save**.

12.2.10. Scanning inactive images

Red Hat Advanced Cluster Security for Kubernetes (RHACS) scans all active (deployed) images every 4 hours and updates the image scan results to reflect the latest vulnerability definitions.

You can also configure RHACS to scan inactive (not deployed) images automatically.

Procedure

1. On the RHACS portal, navigate to **Vulnerability Management > Dashboard**

2. On the **Dashboard** view header, select **IMAGES**.

3. Click **MANAGE WATCHES** to manage the scanning of watched images.

4. In the **MANAGE WATCHED IMAGES** dialog, enter the inactive image's name for which you want to enable scanning.
   Verify that you enter the name of the image and not the image **id**. Image name is the fully-qualified image name, beginning with the registry and ending with the tag. For example, **docker.io/vulnerables/cve-2017-7494:latest**.

5. Select **ADD IMAGE**. RHACS then scans the image and shows the error or success message.

6. (Optional) Click **REMOVE WATCH** to remove an image from the watchlist.

> **IMPORTANT**
>
> On the RHACS portal, click **Platform Configuration > System Configuration** to view the data retention configuration.
>
> All the data related to the image removed from the watchlist continues to appear in the RHACS portal for the number of days mentioned on the **System Configuration** page and is only removed after that period is over.

7. Select **RETURN TO IMAGE LIST** to view the **IMAGES** page.

12.2.11. Creating policies to block specific CVEs

You can create new policies or add specific CVEs to an existing policy from the **Vulnerability Management** view.

Procedure

1. Click **CVEs** from the **Vulnerability Management** view header.

2. Select the checkboxes (leftmost column) for one or more CVEs and then click **Add selected CVEs to Policy** (**add** icon). Or, move the mouse over a CVE in the list, and select the **Add** icon on the right side.

3. For **Policy Name**:

   - To add the CVE to an existing policy, select an existing policy from the drop-down list box.

   - To create a new policy, enter the name for the new policy, and select **Create <policy_name>**.

4. Select a value for **Severity**, either **Critical**, **High**, **Medium**, or **Low**.

5. Choose the **Lifecycle Stage** to which your policy is applicable, from **Build**, or **Deploy**. You can also select both life-cycle stages.

6. Enter details about the policy in the **Description** box.

7. Turn off the **Enable Policy** toggle if you want to create the policy but enable it later. The **Enable Policy** toggle is on by default.

8. Verify the listed CVEs which are included in this policy.

9. Click **Save Policy**.

12.2.12. Viewing recently detected vulnerabilities

The **Recently Detected Vulnerabilities** widget on the **Vulnerability Management** view shows a list of recently discovered vulnerabilities in your scanned images, based on the scan time and CVSS score. It also includes information about the number of images affected by the CVE and its impact (percentage) on your environment.

- When you hover over a CVE in the list, you see an overview of the CVE, which includes scan time, CVSS score, description, impact, and whether it's scored by using CVSS v2 or v3.

- When you select a CVE, the **CVE** details view opens for the selected CVE. The **CVE** details view shows in-depth details of the CVE and the components, images, and deployments and deployments in which it appears.

- Select **View All** on the **Recently Detected Vulnerabilities** widget header to view a list of all the CVEs in your infrastructure. You can also filter the list of CVEs.

12.2.13. Viewing the most common vulnerabilities

The **Most Common Vulnerabilities** widget on the **Vulnerability Management** view shows a list of vulnerabilities that affect the largest number of deployments and images arranged by their CVSS score.

- When you hover over a CVE in the list, you see an overview of the CVE which includes, scan time, CVSS score, description, impact, and whether it is scored by using CVSS v2 or v3.

- When you select a CVE, the **CVE** details view opens for the selected CVE. The **CVE** details view shows in-depth details of the CVE and the components, images, and deployments and deployments in which it appears.

- Select **View All** on the **Most Common Vulnerabilities** widget header to view a list of all the CVEs in your infrastructure. You can also filter the list of CVEs. To export the CVEs as a CSV file, select **Export → Download CVES as CSV**.

12.2.14. Identifying deployments with most severe policy violations

The **Deployments with most severe policy violations** widget on the **Vulnerability Management** view shows a list of deployments and severity of vulnerabilities affecting that deployment.

- When you hover over a deployment in the list, you see an overview of the deployment, which includes the deployment name, the name of the cluster and the namespace in which the deployment exists, and the number of failing policies and their severity.

- When you select a deployment, the **Deployment** view opens for the selected deployment. The **Deployment** view shows in-depth details of the deployment and includes information about policy violations, common vulnerabilities, CVEs, and riskiest images for that deployment.

- Select **View All** on the **Most Common Vulnerabilities** widget header to view a list of all the CVEs in your infrastructure. You can also filter the list of CVEs. To export the CVEs as a CSV file, select **Export → Download CVES as CSV**.

12.2.15. Finding clusters with most Kubernetes and Istio vulnerabilities

Use the **Vulnerability Management** view for identifying the clusters with most Kubernetes and Istio vulnerabilities in your environment.

The **Clusters with most K8S & Istio Vulnerabilities**widget shows a list of clusters, ranked by the number of Kubernetes and Istio vulnerabilities in each cluster. The cluster on top of the list is the cluster with the highest number of vulnerabilities.

Procedure

1. Click on one of the clusters from the list to view details about the cluster. The **Cluster** view includes:

   - **Cluster Details** section, which shows cluster details and metadata, top risky objects (deployments, namespaces, and images), recently detected vulnerabilities, riskiest images, and deployments with the most severe policy violations.

   - **Cluster Findings** section, which includes a list of failing policies and list of fixable CVEs.

   - **Related Entities** section, which shows the number of namespaces, deployments, policies, images, components, and CVEs the cluster contains. You can select these entities to view more details.

2. Click **View All** on the widget header to view the list of all clusters.

12.2.16. Identifying vulnerabilities in nodes

You can use the **Vulnerability Management** view to identify vulnerabilities in your nodes. The identified vulnerabilities include vulnerabilities in:

- Core Kubernetes components.

- Container runtimes (Docker, CRI-O, runC, and containerd).

> **NOTE**
>
> - Red Hat Advanced Cluster Security for Kubernetes does not support identifying vulnerabilities in nodes on OpenShift Container Platform.
>
> - Red Hat Advanced Cluster Security for Kubernetes can identify vulnerabilities in the following operating systems:
>
>   - Amazon Linux 2
>
>   - CentOS
>
>   - Debian
>
>   - Garden Linux (Debian 11)
>
>   - Red Hat Enterprise Linux (RHEL)
>
>   - Ubuntu (AWS, Azure, GCP, and GKE specific versions)

Procedure

1. Select **Nodes** on the **Vulnerability Management** view header to view a list of all the CVEs affecting your nodes.

2. Select a node from the list to view details of all CVEs affecting that node.

   a. When you select a node, the **Node** details panel opens for the selected node. The **Node** view shows in-depth details of the node and includes information about CVEs by CVSS score and fixable CVEs for that node.

   b. Select **View All** on the **CVEs by CVSS score** widget header to view a list of all the CVEs in the selected node. You can also filter the list of CVEs.

   c. To export the fixable CVEs as a CSV file, select **Export as CSV** under the **Node Findings** section.

CHAPTER 13. RESPONDING TO VIOLATIONS

Using Red Hat Advanced Cluster Security for Kubernetes you can view policy violations, drill down to the actual cause of the violation, and take corrective actions.

Red Hat Advanced Cluster Security for Kubernetes built-in policies identify a variety of security findings, including vulnerabilities (CVEs), violations of DevOps best practices, high-risk build and deployment practices, and suspicious runtime behaviors. Whether you use the default out-of-box security policies or use your own custom policies, Red Hat Advanced Cluster Security for Kubernetes reports a violation when an enabled policy fails.

13.1. VIOLATIONS VIEW

You can analyze all violations in the **Violations** view and take corrective action.

To see discovered violations, select **Violations** from the left-hand navigation menu on the RHACS portal.

The **Violations** view shows a list of violations with the following attributes for each row:

- **Deployment**: The name of the deployment.

- **Cluster**: The name of the cluster.

- **Namespace**: The namespace for the deployment.

- **Policy**: The name of the violated policy.

- **Enforced**: Indicates if the policy was enforced when the violation occurred.

- **Severity**: Indicates the severity as **Low**, **Medium**, **High**, or **Critical**.

- **Categories**: The policy categories.

- **Lifecycle**: The lifecycle stages to which the policy applies, **Build**, **Deploy**, or **Runtime**.

- **Time** - The date and time when the violation occurred.

Similar to other views:

- You can select a column heading to sort the violations in ascending or descending order.

- Use the filter bar to filter violations. See the Searching and filtering section for more information.

- Select a violation in the **Violations** view to see more details about the violation.

13.2. VIEWING VIOLATION DETAILS

When you select a violation in the **Violations** view, the **Violation Details** panel opens on the right.

The **Violation Details** panel shows detailed information grouped by multiple tabs.

13.2.1. Violation tab

The **Violation** tab of the **Violation Details** panel explains how the policy was violated. If the policy targets deploy-phase attributes, you can view the specific values that violated the policies, such as violation names. If the policy targets runtime activity, you can view detailed information about the process that violated the policy, including its arguments and the ancestor processes that created it.

13.2.2. Enforcement tab

The **Enforcement** tab of the **Details** panel displays an explanation of the type of enforcement action that was taken in response to the selected policy violation

13.2.3. Deployment tab

The **Deployment** tab of the **Details** panel displays details of the deployment to which the violation applies.

Overview section
The overview section lists the following information:

- **Deployment ID**: The alphanumeric identifier for the deployment.

- **Deployment name**: The name of the deployment.

- **Deployment Type**: The type of the deployment.

- **Cluster**: The name of the cluster where the container is deployed.

- **Replicas**: The number of the replicated deployments.

- **Namespace**: The unique identifier for the deployed cluster.

- **Updated**: The time and date when the deployment was updated.

- **Labels**: The labels that apply to the selected deployment.

- **Annotations**: The annotations that apply to the selected deployment.

- **Service Account**: The name of the service account for the selected deployment.

Container configuration section
The container configuration section lists the following information:

- **Image Name**: The name of the image for the selected deployment.

- **Resources**:

  - **CPU Request (cores)**: The number of cores requested by the container.

  - **Memory Request (MB)**: The memory size requested by the container.

- **Volumes**:

  - **Name**: The name of the location where the service will be mounted.

  - **Source**: The data source path.

  - **Destination**: The path where the data is stored.

- **Type**: The type of the volume.

- **Secrets**: Secrets associated with the selected deployment.

Security context section
Lists whether the container is running as a privileged container.

- **Privileged**:

  - **true** if it is **privileged**.

  - **false** if it is **not privileged**.

Network policy section
Lists all network policies in the namespace containing the violation.

13.2.4. Policy tab

The **Policy** tab of the **Details** panel displays details of the policy that caused the violation.

Policy Details section
The policy details section lists the following information:

- **Id**: The numerical identifier for the policy.

- **Name**: The name of the policy.

- **Description**: A detailed explanation of what the policy alert is about.

- **Rationale**: Information about the reasoning behind the establishment of the policy and why it matters.

- **Remediation**: Suggestions on how to fix the violation.

- **Enabled**: Indicates if the policy is enabled.

- **Categories**: The policy category of the policy.

- **Lifecycle Stage**: Lifecycle stages that the policy belongs to, **Build**, **Deploy**, or **Runtime**.

- **Severity** – The risk level for the violation.

Policy Criteria section
Lists the policy criteria for the policy.

CHAPTER 14. SEARCHING AND FILTERING

The ability to instantly find resources is important to safeguard your cluster. Use Red Hat Advanced Cluster Security for Kubernetes search feature to find relevant resources faster. For example, you can use it to find deployments that are exposed to a newly published CVE or find all deployments that have external network exposure.

14.1. SEARCH SYNTAX

A search query is made up of two parts:

- An attribute that identifies the resource type you want to search for.

- A search term that finds the matching resource.

For example, to find all violations in the **visa-processor** deployment, the search query is **Deployment:visa-processor**. In this search query, **Deployment** is the attribute and **visa-processor** is the search term.

> **NOTE**
>
> You must select an attribute before you can use search terms. However, in some views, such as the **Risk** view and the **Violations** view, Red Hat Advanced Cluster Security for Kubernetes automatically applies the relevant attribute based on the search term you enter.

- You can use multiple attributes in your query. When you use more than one attribute, the results only include the items that match all attributes.

  Example

  When you search for **Namespace:frontend CVE:CVE-2018-11776**, it returns only those resources which violate CVE-2018-11776 in the **frontend** namespace.

- You can use more than one search term with each attribute. When you use more than one search term, the results include all items that match any of the search terms.

  Example

  If you use the search query **Namespace: frontend backend**, it returns matching results from the namespace **frontend** or **backend**.

- You can combine multiple attribute and search term pairs.

  Example

  The search query **Cluster:production Namespace:frontend CVE:CVE-2018-11776** returns all resources which violate CVE-2018-11776 in the **frontend** namespace in the **production** cluster.

- Search terms can be part of a word, in which case Red Hat Advanced Cluster Security for Kubernetes returns all matching results.

  Example

  If you search for **Deployment:def**, the results include all deployments starting with **def**.

- To explicitly search for a specific term, use the search terms inside quotes.

Example

When you search for **Deployment:"def"**, the results only include the deployment **def**.

- You can also use regular expressions by using **r/** before your search term.

  Example

  When you search for **Namespace:r/st.*x**, the results include matches from namespace **stackrox** and **stix**.

- Use **!** to indicate the search terms that you do not want in results.

  Example

  If you search for **Namespace:!stackrox**, the results include matches from all namespaces except the **stackrox** namespace.

- Use the comparison operators **>**, **<**, **=**, **>=**, or **<=** to match a specific value or range of values.

  Example

  If you search for **CVSS:>=6**, the results include all vulnerabilities with Common Vulnerability Scoring System (CVSS) score 6 or higher.

14.2. SEARCH AUTOCOMPLETE

As you enter your query, Red Hat Advanced Cluster Security for Kubernetes automatically displays relevant suggestions for the attributes and the search terms.

14.3. USING GLOBAL SEARCH

By using global search you can search across all resources in your environment. Based on the resource type you use in your search query, the results are grouped in the following categories:

- All (Lists matching results across all categories.)

- Violations

- Policies

- Deployments

- Images

- Secrets

These categories are listed as a table on the RHACS portal global search page and you can click on the category name to identify results belonging to the selected category.

To do a global search, in the RHACS portal, select **Search** on the top right side.

14.4. USING LOCAL PAGE FILTERING

You can use local page filtering from within all views in the RHACS portal. Local page filtering works similar to the global search, but only relevant attributes are available. You can select the search bar to show all available attributes for a specific view.

14.5. COMMON SEARCH QUERIES

Here are some common search queries you can run with Red Hat Advanced Cluster Security for Kubernetes.

Finding deployments that are affected by a specific CVE

Query	Example
CVE:<CVE_number>	CVE:CVE-2018-11776

Finding privileged running deployments

Query	Example
Privileged:<true_or_false>	Privileged:true

Finding deployments that have external network exposure

Query	Example
Exposure Level:<level>	Exposure Level:External

Finding deployments that are running specific processes

Query	Example
Process Name:<process_name>	Process Name:bash

Finding deployments that have serious but fixable vulnerabilities

Query	Example
CVSS:<expression_and_score>	CVSS:>=6 Fixable:.*

Finding deployments that use passwords exposed through environment variables

Query	Example
Environment Key:<query>	Environment Key:r/.*pass.*

Finding running deployments that have particular software components in them

Query	Example
Component:<component_name>	Component:libgpg-error or Component:sudo

Finding users or groups

Use Kubernetes Labels and Selectors, and Annotations to attach metadata to your deployments. You can then query based on the applied annotations and labels to identify individuals or groups.

Finding who owns a particular deployment

Query	Example
Deployment:<deployment_name> Label:<key_value> or **Deployment:<deployment_name> Annotation:<key_value>**	**Deployment:app-server Label:team=backend**

Finding who is deploying images from public registries

Query	Example
Image Registry:<registry_name> Label:<key_value> or **Image Registry:<registry_name> Annotation:<key_value>**	**Image Registry:docker.io Label:team=backend**

Finding who is deploying into the default namespace

Query	Example
Namespace:default Label:<key_value> or **Namespace:default Annotation:<key_value>**	**Namespace:default Label:team=backend**

14.6. SEARCH ATTRIBUTES

Following is the list of search attributes that you can use while searching and filtering in Red Hat Advanced Cluster Security for Kubernetes.

Attribute	Description
Add Capabilities	Provides the container with additional Linux capabilities, for instance the ability to modify files or perform network operations.
Annotation	Arbitrary non-identifying metadata attached to an orchestrator object.
CPU Cores Limit	Maximum number of cores that a resource is allowed to use.
CPU Cores Request	Minimum number of cores to be reserved for a given resource.
CVE	Common Vulnerabilities and Exposures, use it with specific CVE numbers.
CVSS	Common Vulnerability Scoring System, use it with the CVSS score and greater than (>), less than (<), or equal to (=) symbols.

Attribute	Description
Category	Policy categories include DevOps Best Practices, Security Best Practices, Privileges, Vulnerability Management, Multiple, and any custom policy categories that you create.
Cert Expiration	Certificate expiration date.
Cluster	Name of a Kubernetes or OpenShift Container Platform cluster.
Cluster ID	Unique ID for a Kubernetes or OpenShift Container Platform cluster.
Cluster Role	Use **true** to search for cluster-wide roles and **false** for namespace-scoped roles.
Component	Software (daemond, docker), objects (images, containers, services), registries (repository for Docker images).
Component Count	Number of components in the image.
Component version	The version of software, objects, or registries.
Created Time	Time and date when the secret object was created.
Deployment	Name of the deployment.
Deployment Type	The type of Kubernetes controller on which the deployment is based.
Description	Description of the deployment.
Dockerfile Instruction Keyword	Keyword in the Dockerfile instructions in an image.
Dockerfile Instruction Value	Value in the Dockerfile instructions in an image.
Drop Capabilities	Linux capabilities that have been dropped from the container. For example **CAP_SETUID** or **CAP_NET_RAW**.
Enforcement	Type of enforcement assigned to the deployment. For example, **None**, **Scale to Zero Replicas**, or **Add an Unsatisfiable Node Constraint**.
Environment Key	Key portion of a label key-value string that is metadata for further identifying and organizing the environment of a container.
Environment Value	Value portion of a label key-value string that is metadata for further identifying and organizing the environment of a container.

Attribute	Description
Exposed Node Port	Port number of the exposed node port.
Exposing Service	Name of the exposed service.
Exposing Service Port	Port number of the exposed service.
Exposure Level	The type of exposure for a deployment port, for example **external** or **node**.
External Hostname	The hostname for an external port exposure for a deployment.
External IP	The IP address for an external port exposure for a deployment.
Fixable CVE Count	Number of fixable CVEs on an image.
Fixed By	The version string of a package that fixes a flagged vulnerability in an image.
Image	The name of the image.
Image Command	The command specified in the image.
Image Created Time	The time and date when the image was created.
Image Entrypoint	The entrypoint command specified in the image.
Image Pull Secret	The name of the secret to use when pulling the image, as specified in the deployment.
Image Pull Secret Registry	The name of the registry for an image pull secret.
Image Registry	The name of the image registry.
Image Remote	Indication of an image that is remotely accessible.
Image Scan Time	The time and date when the image was last scanned.
Image Tag	Identifier for an image.
Image Users	Name of the user or group that a container image is configured to use when it runs.
Image Volumes	Names of the configured volumes in the container image.
Inactive Deployment	Use **true** to search for inactive deployments and **false** for active deployments.

Attribute	Description
Label	The key portion of a label key-value string that is metadata for further identifying and organizing images, containers, daemons, volumes, networks, and other resources.
Lifecycle Stage	The type of lifecycle stage where this policy is configured or alert was triggered.
Max Exposure Level	For a deployment, the maximum level of network exposure for all given ports/services.
Memory Limit (MB)	Maximum amount of memory that a resource is allowed to use.
Memory Request (MB)	Minimum amount of memory to be reserved for a given resource.
Namespace	The name of the namespace.
Namespace ID	Unique ID for the containing namespace object on a deployment.
Node	Name of a node.
Node ID	Unique ID for a node.
Pod Label	Single piece of identifying metadata attached to an individual pod.
Policy	The name of the security policy.
Port	Port numbers exposed by a deployment.
Port Protocol	IP protocol such as TCP or UDP used by exposed port.
Priority	Risk priority for a deployment. (Only available in **Risks** view.)
Privileged	Use **true** to search for privileged running deployments, or **false** otherwise.
Process Ancestor	Name of any parent process for a process indicator in a deployment.
Process Arguments	Command arguments for a process indicator in a deployment.
Process Name	Name of the process for a process indicator in a deployment.
Process Path	Path to the binary in the container for a process indicator in a deployment.
Process UID	Unix user ID for the process indicator in a deployment.
Read Only Root Filesystem	Use **true** to search for containers running with the root file system configured as read only.

Attribute	Description
Role	Name of a Kubernetes RBAC role.
Role Binding	Name of a Kubernetes RBAC role binding.
Role ID	Role ID to which a Kubernetes RBAC role binding is bound.
Secret	Name of the secret object that holds the sensitive information.
Secret Path	Path to the secret object in the file system.
Secret Type	Type of the secret, for example, certificate or RSA public key.
Service Account	Service account name for a service account or deployment.
Severity	Indication of level of importance of a violation: Critical, High, Medium, Low.
Subject	Name for a subject in Kubernetes RBAC.
Subject Kind	Type of subject in Kubernetes RBAC, such as **SERVICE_ACCOUNT**, **USER** or **GROUP**.
Taint Effect	Type of taint currently applied to a node.
Taint Key	Key for a taint currently applied to a node.
Taint Value	Allowed value for a taint currently applied to a node.
Toleration Key	Key for a toleration applied to a deployment.
Toleration Value	Value for a toleration applied to a deployment.
Violation	A notification displayed in the **Violations** page when the conditions specified by a policy have not been met.
Violation State	Use it to search for resolved violations.
Violation Time	Time and date that a violation first occurred.
Volume Destination	Mount path of the data volume.
Volume Name	Name of the storage.
Volume ReadOnly	Use **true** to search for volumes that are mounted as read only.

Attribute	Description
Volume Source	Indicates the form in which the volume is provisioned (for example, **persistentVolumeClaim** or **hostPath**).
Volume Type	The type of volume.

CHAPTER 15. MANAGING USER ACCESS

15.1. CONFIGURING OKTA IDENTITY CLOUD AS A SAML 2.0 IDENTITY PROVIDER

You can use Okta as a single sign-on (SSO) provider for Red Hat Advanced Cluster Security for Kubernetes (RHACS).

15.1.1. Creating an Okta app

Before you can use Okta as a SAML 2.0 identity provider for Red Hat Advanced Cluster Security for Kubernetes, you must create an Okta app.

> **WARNING**
>
> Okta's **Developer Console** does not support the creation of custom SAML 2.0 applications. If you are using the **Developer Console**, you must first switch to the **Admin Console** (**Classic UI**). To switch, click **Developer Console** in the top left of the page and select **Classic UI**.

Prerequisites

- You must have an account with administrative privileges for the Okta portal.

Procedure

1. On the Okta portal, select **Applications** from the menu bar.

2. Click **Add Application** and then select **Create New App**.

3. In the **Create a New Application Integration** dialog box, leave **Web** as the platform and select **SAML 2.0** as the protocol that you want to sign in users.

4. Click **Create**.

5. On the **General Settings** page, enter a name for the app in the **App name** field.

6. Click **Next**.

7. On the **SAML Settings** page, set values for the following fields:

   a. **Single sign on URL**

      - Specify it as **https://<RHACS_portal_hostname>/sso/providers/saml/acs**.

      - Leave the **Use this for Recipient URL and Destination URL** option checked.

      - If your RHACS portal is accessible at different URLs, you can add them here by checking the **Allow this app to request other SSO URLs** option and add the alternative URLs using the specified format.

    b. **Audience URI (SP Entity ID)**

- Set the value to **RHACS** or another value of your choice.

- Remember the value you choose; you will need this value when you configure Red Hat Advanced Cluster Security for Kubernetes.

    c. **Attribute Statements**

- You must add at least one attribute statement.

- Red Hat recommends using the email attribute:

  - **Name:** email

  - **Format:** Unspecified

  - **Value:** user.email

8. Verify that you have configured at least one **Attribute Statement** before continuing.

9. Click **Next**.

10. On the **Feedback** page, select an option that applies to you.

11. Select an appropriate **App type**.

12. Click **Finish**.

After the configuration is complete, you are redirected to the **Sign On** settings page for the new app. A yellow box contains links to the information that you need to configure Red Hat Advanced Cluster Security for Kubernetes.

After you have created the app, assign Okta users to this application. Go to the **Assignments** tab, and assign the set of individual users or groups that can access Red Hat Advanced Cluster Security for Kubernetes. For example, assign the group **Everyone** to allow all users in the organization to access Red Hat Advanced Cluster Security for Kubernetes.

15.1.2. Configuring a SAML 2.0 identity provider

Use the instructions in this section to integrate a Security Assertion Markup Language (SAML) 2.0 identity provider with Red Hat Advanced Cluster Security for Kubernetes (RHACS).

Prerequisites

- You must have permissions to configure identity providers in RHACS.

- For Okta identity providers, you must have an Okta app configured for RHACS.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Click **Create auth provider** and select **SAML 2.0** from the drop-down list.

3. In the **Name** field, enter a name to identify this authentication provider; for example, **Okta** or **Google**. The integration name is shown on the login page to help users select the correct sign-in option.

4. In the **ServiceProvider issuer** field, enter the value that you are using as the **Audience URI** or **SP Entity ID** in Okta, or a similar value in other providers.

5. Select the type of **Configuration**:

   - **Option 1: Dynamic Configuration**: If you select this option, enter the **IdP Metadata URL,** or the URL of **Identity Provider metadata** available from your identity provider console. The configuration values are acquired from the URL.

   - **Option 2: Static Configuration**: Copy the required static fields from the **View Setup Instructions** link in the Okta console, or a similar location for other providers:

     - IdP Issuer

     - IdP SSO URL

     - Name/ID Format

     - IdP Certificate(s) (PEM)

6. Assign a **Minimum access role** for users who access RHACS using SAML.

   TIP

   Set the **Minimum access role** to **Admin** while you complete setup. Later, you can return to the **Access Control** page to set up more tailored access rules based on user metadata from your identity provider.

7. Click **Save**.
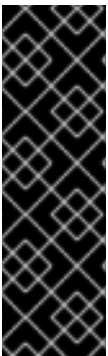
   IMPORTANT

   If your SAML identity provider's authentication response meets the following criteria:

   - Includes a **NotValidAfter** assertion: The user session remains valid until the time specified in the **NotValidAfter** field has elapsed. After the user session expires, users must reauthenticate.

   - Does not include a **NotValidAfter** assertion: The user session remains valid for 30 days, and then users must reauthenticate.

Verification

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Select the **Auth Providers** tab.

3. Click the authentication provider for which you want to verify the configuration.

4. Select **Test login** from the **Auth Provider** section header. The **Test login** page opens in a new browser tab.

5. Sign in with your credentials.

- If you logged in successfully, RHACS shows the **User ID** and **User Attributes** that the identity provider sent for the credentials that you used to log in to the system.

- If your login attempt failed, RHACS shows a message describing why the identity provider's response could not be processed.

6. Close the **Test login** browser tab.

> **NOTE**
>
> Even if the response indicates successful authentication, you might need to create additional access rules based on the user metadata from your identity provider.

15.2. CONFIGURING GOOGLE WORKSPACE AS AN OIDC IDENTITY PROVIDER

You can use Google Workspace as a single sign-on (SSO) provider for Red Hat Advanced Cluster Security for Kubernetes.

15.2.1. Setting up OAuth 2.0 credentials for your GCP project

To configure Google Workspace as an identity provider for Red Hat Advanced Cluster Security for Kubernetes, you must first configure OAuth 2.0 credentials for your GCP project.

Prerequisites

- You must have administrator-level access to your organization's Google Workspace account to create a new project, or permissions to create and configure OAuth 2.0 credentials for an existing project. Red Hat recommends that you create a new project for managing access to Red Hat Advanced Cluster Security for Kubernetes.

Procedure

1. Create a new Google Cloud Platform (GCP) project, see the Google documentation topic creating and managing projects.

2. After you have created the project, open the Credentials page in the Google API Console.

3. Verify the project name listed in the upper left corner near the logo to make sure that you are using the correct project.

4. To create new credentials, go to **Create Credentials → OAuth client ID**.

5. Choose **Web application** as the **Application type**.

6. In the **Name** box, enter a name for the application, for example, RHACS.

7. In the **Authorized redirect URIs** box, enter **https://<stackrox_hostname>:<port_number>/sso/providers/oidc/callback**.

- replace **<stackrox_hostname>** with the hostname on which you expose your Central instance.

- replace **<port_number>** with the port number on which you expose Central. If you are using the standard HTTPS port **443**, you can omit the port number.

8. Click **Create**. This creates an application and credentials and redirects you back to the credentials page.

9. An information box opens, showing details about the newly created application. Close the information box.

10. Copy and save the **Client ID** that ends with **.apps.googleusercontent.com**. You can check this client ID by using the Google API Console.

11. Select **OAuth consent screen** from the navigation menu on the left.

> **NOTE**
>
> The OAuth consent screen configuration is valid for the entire GCP project, and not only to the application you created in the previous steps. If you already have an OAuth consent screen configured in this project and want to apply different settings for Red Hat Advanced Cluster Security for Kubernetes login, create a new GCP project.

12. On the OAuth consent screen page:

    a. Choose the **Application type** as **Internal**. If you select **Public**, anyone with a Google account can sign in.

    b. Enter a descriptive **Application name**. This name is shown to users on the consent screen when they sign in. For example, use **RHACS** or **<organization_name> SSO for Red Hat Advanced Cluster Security for Kubernetes**.

    c. Verify that the **Scopes for Google APIs** only lists **email**, **profile**, and **openid** scopes. Only these scopes are required for single sign-on. If you grant additional scopes it increases the risk of exposing sensitive data.

15.2.2. Specifying a client secret

Red Hat Advanced Cluster Security for Kubernetes version 3.0.39 and newer supports the OAuth 2.0 Authorization Code Grant authentication flow when you specify a client secret. When you use this authentication flow, Red Hat Advanced Cluster Security for Kubernetes uses a refresh token to keep users logged in beyond the token expiration time configured in your OIDC identity provider.

When users log out, Red Hat Advanced Cluster Security for Kubernetes deletes the refresh token from the client-side. Additionally, if your identity provider API supports refresh token revocation, Red Hat Advanced Cluster Security for Kubernetes also sends a request to your identity provider to revoke the refresh token.

You can specify a client secret when you configure Red Hat Advanced Cluster Security for Kubernetes to integrate with an OIDC identity provider.

> **NOTE**
>
> - You cannot use a **Client Secret** with the *Fragment* **Callback mode**.
>
> - You cannot edit configurations for existing authentication providers.
>
> - You must create a new OIDC integration in Red Hat Advanced Cluster Security for Kubernetes if you want to use a **Client Secret**.

Red Hats recommends using a client secret when connecting Red Hat Advanced Cluster Security for Kubernetes with an OIDC identity provider. If you do not want to use a **Client Secret**, you must select the **Do not use Client Secret (not recommended)** option.

15.2.3. Configuring an OIDC identity provider

You can configure Red Hat Advanced Cluster Security for Kubernetes (RHACS) to use your OpenID Connect (OIDC) identity provider.

Prerequisites

- You must have already configured an application in your identity provider, such as Google Workspace.

- You must have permissions to configure identity providers in RHACS.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Click **Create auth provider** and select **OpenID Connect** from the drop-down list.

3. Enter information in the following fields:

   - **Name**: A name to identify your authentication provider; for example, **Google Workspace**. The integration name is shown on the login page to help users select the correct sign-in option.

   - **Callback mode**: Select **Auto-select (recommended)**, which is the default value, unless the identity provider requires another mode.

     > **NOTE**
     >
     > **Fragment** mode is designed around the limitations of Single Page Applications (SPAs). Red Hat only supports the **Fragment** mode for early integrations and does not recommended using it for later integrations.

   - **Issuer**: The root URL of your identity provider; for example, **https://accounts.google.com** for Google Workspace. See your identity provider documentation for more information.

NOTE

If you are using RHACS version 3.0.49 and later, for **Issuer** you can perform these actions:

- Prefix your root URL with **https+insecure://** to skip TLS validation. This configuration is insecure and Red Hat does not recommended it. Only use it for testing purposes.

- Specify query strings; for example, **?key1=value1&key2=value2** along with the root URL. RHACS appends the value of **Issuer** as you entered it to the authorization endpoint. You can use it to customize your provider's login screen. For example, you can optimize the Google Workspace login screen to a specific hosted domain by using the **hd parameter**, or preselect an authentication method in **PingFederate** by using the **pfidpadapterid** parameter.

- **Client ID**: The OIDC Client ID for your configured project.

- **Client Secret**: Enter the client secret provided by your identity provider (IdP). If you are not using a client secret, which is not recommended, select **Do not use Client Secret**.

4. Assign a **Minimum access role** for users who access RHACS using the selected identity provider.

TIP

Set the **Minimum access role** to **Admin** while you complete setup. Later, you can return to the **Access Control** page to set up more tailored access rules based on user metadata from your identity provider.

5. To add access rules for users and groups accessing RHACS, click **Add new rule** in the **Rules** section. For example, to give the **Admin** role to a user called **administrator**, you can use the following key–value pairs to create access rules:

Key	Value
Name	administrator
Role	Admin

6. Click **Save**.

Verification

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Select the **Auth providers** tab.

3. Select the authentication provider for which you want to verify the configuration.

4. Select **Test login** from the **Auth Provider** section header. The **Test login** page opens in a new browser tab.

5. Log in with your credentials.

   - If you logged in successfully, RHACS shows the **User ID** and **User Attributes** that the identity provider sent for the credentials that you used to log in to the system.

   - If your login attempt failed, RHACS shows a message describing why the identity provider's response could not be processed.

6. Close the **Test Login** browser tab.

15.3. CONFIGURING OPENSHIFT CONTAINER PLATFORM OAUTH SERVER AS AN IDENTITY PROVIDER

OpenShift Container Platform includes a built-in OAuth server that you can use as an authentication provider for Red Hat Advanced Cluster Security for Kubernetes (RHACS).

15.3.1. Configuring OpenShift Container Platform OAuth server as an identity provider

To integrate the built-in OpenShift Container Platform OAuth server as an identity provider for RHACS, use the instructions in this section.

Prerequisites

- You must have the **AuthProvider** permission to configure identity providers in RHACS.

- You must have already configured users and groups in OpenShift Container Platform OAuth server through an identity provider. For information about the identity provider requirements, see Understanding identity provider configuration .

> **NOTE**
>
> The following procedure configures only a single main route named **central** for the OpenShift Container Platform OAuth server.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Click **Create auth provider** and select **OpenShift Auth** from the drop-down list.

3. Enter a name for the authentication provider in the **Name** field.

4. Assign a **Minimum access role** for users that access RHACS using the selected identity provider. A user must have the permissions granted to this role or a role with higher permissions to log in to RHACS.

   > **TIP**
   >
   > For security, Red Hat recommends first setting the **Minimum access role** to **None** while you complete setup. Later, you can return to the **Access Control** page to set up more tailored access rules based on user metadata from your identity provider.

5. Optional: To add access rules for users and groups accessing RHACS, click **Add new rule** in the **Rules** section, then enter the rule information and click **Save**. You will need attributes for the user or group so that you can configure access.

TIP

Group mappings are more robust because groups are usually associated with teams or permissions sets and require modification less often than users.

To get user information in OpenShift Container Platform, you can use one of the following methods:

- Click **User Management** → **Users** → **<username>** → YAML.

- Access the **k8s/cluster/user.openshift.io~v1~User/<username>/yaml** file and note the values for **name**, **uid** (**userid** in RHACS), and **groups**.

- Use the OpenShift Container Platform API as described in the *OpenShift Container Platform API reference*.

The following configuration example describes how to configure rules for an **Admin** role with the following attributes:

- **name**: **administrator**

- **groups**: **["system:authenticated", "system:authenticated:oauth", "myAdministratorsGroup"]**

- **uid**: **12345-00aa-1234-123b-123fcdef1234**

You can add a rule for this administrator role using one of the following steps:

- To configure a rule for a name, select **name** from the **Key** drop-down list, enter **administrator** in the **Value** field, then select **Administrator** under **Role**.

- To configure a rule for a group, select **groups** from the **Key** drop-down list, enter **myAdministratorsGroup** in the **Value** field, then select **Admin** under **Role**.

- To configure a rule for a user name, select **userid** from the **Key** drop-down list, enter **12345-00aa-1234-123b-123fcdef1234** in the **Value** field, then select **Admin** under **Role**.

> **IMPORTANT**
>
> - If you use a custom TLS certificate for OpenShift Container Platform OAuth server, you must add the root certificate of the CA to Red Hat Advanced Cluster Security for Kubernetes as a trusted root CA. Otherwise, Central cannot connect to the OpenShift Container Platform OAuth server.
>
> - To enable the OpenShift Container Platform OAuth server integration when installing Red Hat Advanced Cluster Security for Kubernetes using the **roxctl** CLI, set the **ROX_ENABLE_OPENSHIFT_AUTH** environment variable to **true** in Central:
>
>   ```
>   $ oc -n stackrox set env deploy/central
>   ROX_ENABLE_OPENSHIFT_AUTH=true
>   ```
>
> - For access rules, the OpenShift Container Platform OAuth server does not return the key **Email**.

Additional resources

- [Configuring an LDAP identity provider](#)

- [Adding trusted certificate authorities](#)

15.3.2. Creating additional routes for OpenShift Container Platform OAuth server

When you configure OpenShift Container Platform OAuth server as an identity provider by using Red Hat Advanced Cluster Security for Kubernetes portal, RHACS configures only a single route for the OAuth server. However, you can create additional routes by specifying them as annotations in the Central custom resource.

Prerequisites

- You must have configured [Service accounts as OAuth clients](#) for your OpenShift Container Platform OAuth server.

Procedure

- If you installed RHACS using the RHACS Operator:

    1. Create a **CENTRAL_ADDITIONAL_ROUTES** environment variable that contains a patch for the Central custom resource:

        ```
        $ CENTRAL_ADDITIONAL_ROUTES='
        spec:
          central:
            exposure:
              loadBalancer:
                enabled: false
                port: 443
              nodePort:
                enabled: false
              route:
                enabled: true
            persistence:
        ```

```
      persistentVolumeClaim:
        claimName: stackrox-db
   customize:
    annotations:
      serviceaccounts.openshift.io/oauth-redirecturi.main: sso/providers/openshift/callback
```
1
```
      serviceaccounts.openshift.io/oauth-redirectreference.main: "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"central\"}}"
```
2
```
      serviceaccounts.openshift.io/oauth-redirecturi.second:
sso/providers/openshift/callback
```
3
```
      serviceaccounts.openshift.io/oauth-redirectreference.second: "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"second-central\"}}"
```
4
```
'
```

1 The redirect URI for setting the main route.

2 The redirect URI reference for the main route.

3 The redirect for setting the second route.

4 The redirect reference for the second route.

2. Apply the **CENTRAL_ADDITIONAL_ROUTES** patch to the Central custom resource:

```
$ oc patch centrals.platform.stackrox.io \
  -n <namespace> \
  <custom-resource> \
  --patch "$CENTRAL_ADDITIONAL_ROUTES" \
  --type=merge
```
1
2

1 Replace **<namespace>** with the name of the project that contains the Central custom resource.

2 Replace **<custom-resource>** with the name of the Central custom resource.

- Or, if you installed RHACS using Helm:

    1. Add the following annotations to your **values-public.yaml** file:

```
customize:
 central:
   annotations:
     serviceaccounts.openshift.io/oauth-redirecturi.main: sso/providers/openshift/callback
```
1
```
     serviceaccounts.openshift.io/oauth-redirectreference.main: "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"central\"}}"
```
2
```
     serviceaccounts.openshift.io/oauth-redirecturi.second:
sso/providers/openshift/callback
```
3

```
        serviceaccounts.openshift.io/oauth-redirectreference.second: "
{\"kind\":\"OAuthRedirectReference\",\"apiVersion\":\"v1\",\"reference\":
{\"kind\":\"Route\",\"name\":\"second-central\"}}" 4
```

1 The redirect for setting the main route.

2 The redirect reference for the main route.

3 The redirect for setting the second route.

4 The redirect reference for the second route.

2. Apply the custom annotations to the Central custom resource by using **helm upgrade**:

```
$ helm upgrade -n stackrox \
  stackrox-central-services rhacs/central-services \
  -f <path_to_values_public.yaml> 1
```

1 Specify the path of the **values-public.yaml** configuration file using the **-f** option.

Additional resources

- [Service accounts as OAuth clients](#)

- [Redirect URIs for service accounts as OAuth clients](#)

15.4. MANAGING RBAC IN RED HAT ADVANCED CLUSTER SECURITY FOR KUBERNETES

Red Hat Advanced Cluster Security for Kubernetes (RHACS) comes with role-based access control (RBAC) that you can use to configure roles and grant various levels of access to Red Hat Advanced Cluster Security for Kubernetes for different users.

Red Hat Advanced Cluster Security for Kubernetes 3.63 includes a scoped access control feature that enables you to configure fine-grained and specific sets of permissions that define how a given Red Hat Advanced Cluster Security for Kubernetes user or a group of users can interact with Red Hat Advanced Cluster Security for Kubernetes, which resources they can access, and which actions they can perform.

- *Roles* are a collection of permission sets and access scopes. You can assign roles to users and groups by specifying rules. You can configure these rules when you configure an authentication provider. There are two types of roles in Red Hat Advanced Cluster Security for Kubernetes:

  - System roles that are created by Red Hat and cannot be changed.

  - Custom roles, which Red Hat Advanced Cluster Security for Kubernetes administrators can create and change at any time.

NOTE

- If you assign multiple roles for a user, they get access to the combined permissions of the assigned roles.

- If you have users assigned to a custom role, and you delete that role, all associated users transfer to the minimum access role that you have configured.

- *Permission sets* are a set of permissions that define what actions a role can perform on a given resource. *Resources* are the functionalities of Red Hat Advanced Cluster Security for Kubernetes for which you can set view (**read**) and modify (**write**) permissions. There are two types of permission sets in Red Hat Advanced Cluster Security for Kubernetes:

  - System permission sets, which are created by Red Hat and cannot be changed.

  - Custom permission sets, which Red Hat Advanced Cluster Security for Kubernetes administrators can create and change at any time.

- *Access scopes* are a set of Kubernetes and OpenShift Container Platform resources that users can access. For example, you can define an access scope that only allows users to access information about pods in a given project. There are two types of access scopes in Red Hat Advanced Cluster Security for Kubernetes:

  - System access scopes, which are created by Red Hat and cannot be changed.

  - Custom access scopes, which Red Hat Advanced Cluster Security for Kubernetes administrators can create and change at any time.

15.4.1. System roles

Red Hat Advanced Cluster Security for Kubernetes (RHACS) includes some default system roles that you can apply to users when you create rules. You can also create custom roles as required.

System role	Description
Admin	This role is targeted for administrators. Use it to provide read and write access to all resources.
Analyst	This role is targeted for a user who cannot make any changes, but can view everything. Use it to provide read-only access for all resources.
Continuous Integration	This role is targeted for CI (continuous integration) systems and includes the permission set required to enforce deployment policies.
None	This role has no read and write access to any resource. You can set this role as the minimum access role for all users.
Sensor Creator	RHACS uses this role to automate new cluster setups. It includes the permission set to create Sensors in secured clusters.
Scope Manager	This role includes the minimum permissions required to create and modify access scopes.

System role	Description
Vulnerability Management Approver	This role allows you to provide access to approve vulnerability deferrals or false positive requests.
Vulnerability Management Requester	This role allows you to provide access to request vulnerability deferrals or false positives.
Vulnerability Report Creator	This role allows you to create and manage vulnerability reporting configurations for scheduled vulnerability reports.

15.4.1.1. Viewing the permission set and access scope for a system role

You can view the permission set and access scope for the default system roles.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access control**.

2. Select **Roles**.

3. Click on one of the roles to view its details. The details page shows the permission set and access scope for the slected role.

> **NOTE**
>
> You cannot modify permission set and access scope for the default system roles.

15.4.1.2. Creating a custom role

You can create new roles from the **Access Control** view.

Prerequisites

- You must have the **Admin** role, or a role with the permission set with read and write permissions for the **AuthProvider** and **Role** resources to create, modify, and delete custom roles.

- You must create a permissions set and an access scope for the custom role before creating the role.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Select **Roles**.

3. Click **Create role**.

4. Enter a **Name** and **Description** for the new role.

5. Select a **Permission set** for the role.

6. Select an **Access scope** for the role.

7. Click **Save**.

Additional resources

- Creating a custom permission set

- Creating a custom access scope

15.4.1.3. Assigning a role to a user or a group

You can use the RHACS portal to assign roles to a user or a group.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. From the list of authentication providers, select the authentication provider.

3. Click **Edit minimum role and rules**

4. Under the **Rules** section, click **Add new rule**.

5. For **Key**, select one of the values from **userid**, **name**, **email** or **group**.

6. For **Value**, enter the value of the user ID, name, email address or group based on the key you selected.

7. Click the **Role** drop-down menu and select the role you want to assign.

8. Click **Save**.

You can repeat these instructions for each user or group and assign different roles.

15.4.2. System permission sets

Red Hat Advanced Cluster Security for Kubernetes includes some default system permission sets that you can apply to roles. You can also create custom permission sets as required.

Permission set	Description
Admin	Provides read and write access to all resources.
Analyst	Provides read-only access for all resources.
Continuous Integration	This permission set is targeted for CI (continuous integration) systems and includes the permissions required to enforce deployment policies.
None	No read and write permissions are allowed for any resource.
Sensor Creator	Provides permissions for resources that are required to create Sensors in secured clusters.

15.4.2.1. Viewing the permissions for a system permission set

You can view the permissions for a system permission set in the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access control**.

2. Select **Permission sets**.

3. Click on one of the permission sets to view its details. The details page shows a list of resources and their permissions for the selected permission set.

> **NOTE**
>
> You cannot modify permissions for a system permission set.

15.4.2.2. Creating a custom permission set

You can create new permission sets from the **Access Control** view.

Prerequisites

- You must have the **Admin** role, or a role with the permission set with read and write permissions for the **AuthProvider** and **Role** resources to create, modify, and delete permission sets.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Select **Permission sets**.

3. Click **Create permission set**

4. Enter a **Name** and **Description** for the new permission set.

5. For each resource, under the **Access level** column, select one of the permissions from **No access**, **Read access**, or **Read and Write access**.

> **WARNING**
>
> - If you are configuring a permission set for users, you must grant read-only permissions for the following resources:
>
>   - **Alert**
>
>   - **Cluster**
>
>   - **Deployment**
>
>   - **Image**
>
>   - **NetworkPolicy**
>
>   - **NetworkGraph**
>
>   - **Policy**
>
>   - **Secret**
>
> - These permissions are preselected when you create a new permission set.
>
> - If you do not grant these permissions, users will experience issues with viewing pages in the RHACS portal.

6. Click **Save**.

15.4.3. System access scopes

Red Hat Advanced Cluster Security for Kubernetes includes some default system access scopes that you can apply on roles. You can also create custom access scopes as required.

Acces scope	Description
Unrestricted	Provides access to all clusters and namespaces that Red Hat Advanced Cluster Security for Kubernetes monitors.
Deny All	Provides no access to any Kubernetes and OpenShift Container Platform resources.

15.4.3.1. Viewing the details for a system access scope

You can view the Kubernetes and OpenShift Container Platform resources that are allowed and not allowed for an access scope in the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access control**.

2. Select **Access scopes**.

3. Click on one of the access scopes to view its details. The details page shows a list of clusters and namespaces, and which ones are allowed for the selected access scope.

> **NOTE**
>
> You cannot modify allowed resources for a system access scope.

15.4.3.2. Creating a custom access scope

You can create new access scopes from the **Access Control** view.

Prerequisites

- You must have the **Admin** role, or a role with the permission set with read and write permissions for the **AuthProvider** and **Role** resources to create, modify, and delete permission sets.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access control**.

2. Select **Access scopes**.

3. Click **Create access scope**.

4. Enter a **Name** and **Description** for the new access scope.

5. Under the **Allowed resources** section:

   - Use the **Cluster filter** and **Namespace filter** fields to filter the list of clusters and namespaces visible in the list.

   - Expand the **Cluster name** to see the list of namespaces in that cluster.

   - To allow access to all namespaces in a cluster, toggle the switch in the **Manual selection** column.

     > **NOTE**
     >
     > Access to a specific cluster provides users with access to the following resources within the scope of the cluster:
     >
     > - OpenShift Container Platform or Kubernetes cluster metadata and security information
     >
     > - Compliance information for authorized clusters
     >
     > - Node metadata and security information
     >
     > - Access to all namespaces in that cluster and their associated security information

- To allow access to a namespace, toggle the switch in the **Manual selection** column for a namespace.

> **NOTE**
>
> Access to a specific namespace gives access to the following information within the scope of the namespace:
>
> - Alerts and violations for deployments
>
> - Vulnerability data for images
>
> - Deployment metadata and security information
>
> - Role and user information
>
> - Network graph, policy, and baseline information for deployments
>
> - Process information and process baseline configuration
>
> - Prioritized risk information for each deployment

6. If you want to allow access to clusters and namespaces based on labels, click **Add label selector** under the **Label selection rules** section. Then click **Add rule** to specify **Key** and **Value** pairs for the label selector. You can specify labels for clusters and namespaces.

7. Click **Save**.

15.4.4. Resource definitions

Red Hat Advanced Cluster Security for Kubernetes includes multiple resources. The following table lists the resources and describes the actions that users can perform with the **read** or **write** permission.

Resource	Read permission	Write permission
APIToken	List existing API tokens.	Create new API tokens or revoke existing tokens.
Alert	View existing policy violations.	Resolve or edit policy violations.
AuthPlugin	View existing Authentication plugins	Modify these configurations. *(Local administrator only.)*
AuthProvider	View existing configurations for single sign-on.	Modify these configurations.
BackupPlugins	View existing integrations with automated backup systems such as AWS S3.	Modify these configurations.
CVE	*Internal use only*	*Internal use only*

Resource	Read permission	Write permission
Cluster	View existing secured clusters.	Add new secured clusters and modify or delete existing clusters.
Compliance	View compliance standards and results.	N/A
ComplianceRunSchedule	View scheduled compliance runs.	Create, modify, or delete scheduled compliance runs.
ComplianceRuns	View recent compliance runs and their completion status.	Trigger compliance runs.
Config	View options for data retention, security notices, and other related configurations.	Modify these configurations.
DebugLogs	View the current logging verbosity level in Red Hat Advanced Cluster Security for Kubernetes components.	Modify the logging level.
Deployment	View deployments (workloads) in secured clusters.	N/A
Detection	Check build-time policies against images or deployment YAML.	N/A
Group	View the existing RBAC rules that match user metadata to Red Hat Advanced Cluster Security for Kubernetes roles.	Create, modify, or delete configured RBAC rules.
Image	View images, their components, and their vulnerabilities.	N/A
ImageComponent	*Internal use only*	*Internal use only*
ImageIntegration	List existing image registry integrations.	Create, edit, or delete image registry integrations.
ImbuedLogs	*Internal use only*	*Internal use only*
Indicator	View process activity in deployments.	N/A
K8sRole	View roles for Kubernetes role-based access control in secured clusters.	N/A
K8sRoleBinding	View role bindings for Kubernetes role-based access control in secured clusters.	N/A

Resource	Read permission	Write permission
K8sSubject	View users and groups for Kubernetes role-based access control in secured clusters.	N/A
Licenses	View the status of the existing license for Red Hat Advanced Cluster Security for Kubernetes.	Upload a new license key.
Namespace	View existing Kubernetes namespaces in secured clusters.	N/A
NetworkGraph	View active and allowed network connections in secured clusters.	N/A
NetworkPolicy	View existing network policies in secured clusters and simulate changes.	Apply network policy changes in secured clusters.
Node	View existing Kubernetes nodes in secured clusters.	N/A
Notifier	View existing integrations for notification systems like email, Jira, or webhooks.	Create, modify, or delete these integrations.
Policy	View existing system policies.	Create, modify, or delete system policies.
ProbeUpload	Read manifests for the uploaded probe files.	Upload support packages to Central.
ProcessWhitelist	View process baselines.	Add or remove processes from baselines.
Risk	View Risk results.	N/A
Role	View existing Red Hat Advanced Cluster Security for Kubernetes RBAC roles and their permissions.	Add, modify, or delete roles and their permissions.
ScannerBundle	Download the scanner bundle.	N/A
ScannerDefinitions	List existing image scanner integrations.	Create, modify, or delete image scanner integrations.
Secret	View metadata about secrets in secured clusters.	N/A
SensorUpgradeConfig	Check the status of automatic upgrades.	Disable or enable automatic upgrades for secured clusters.

Resource	Read permission	Write permission
ServiceAccount	List Kubernetes service accounts in secured clusters.	N/A
ServiceIdentity	View metadata about Red Hat Advanced Cluster Security for Kubernetes service-to-service authentication.	Revoke or reissue service-to-service authentication credentials.
User	View users that have accessed your Red Hat Advanced Cluster Security for Kubernetes instance, including the metadata that the authentication provider provides about them.	N/A

15.5. ENABLING PKI AUTHENTICATION

If you use an enterprise certificate authority (CA) for authentication, you can configure Red Hat Advanced Cluster Security for Kubernetes (RHACS) to authenticate users by using their personal certificates.

After you configure PKI authentication, users and API clients can log in using their personal certificates. Users without certificates can still use other authentication options, including API tokens, the local administrator password, or other authentication providers. PKI authentication is available on the same port number as the Web UI, gRPC, and REST APIs.

When you configure PKI authentication, by default, Red Hat Advanced Cluster Security for Kubernetes uses the same port for PKI, web UI, gRPC, other single sign-on (SSO) providers, and REST APIs. You can also configure a separate port for PKI authentication by using a YAML configuration file to configure and expose endpoints.

15.5.1. Configuring PKI authentication by using the RHACS portal

You can configure Public Key Infrastructure (PKI) authentication by using the RHACS portal.

Procedure

1. On the RHACS portal, navigate to **Platform Configuration → Access Control**.

2. Click **Create Auth Provider** and select **User Certificates** from the drop-down list.

3. In the **Name** field, specify a name for this authentication provider.

4. In the **CA certificate(s) (PEM)** field, paste your root CA certificate in PEM format.

5. Assign a **Minimum access role** for users who access RHACS using PKI authentication. A user must have the permissions granted to this role or a role with higher permissions to log in to RHACS.

TIP

For security, Red Hat recommends first setting the **Minimum access role** to **None** while you complete setup. Later, you can return to the **Access Control** page to set up more tailored access rules based on user metadata from your identity provider.

6. To add access rules for users and groups accessing RHACS, click **Add new rule** in the **Rules** section. For example, to give the **Admin** role to a user called **administrator**, you can use the following key-value pairs to create access rules:

Key	Value
Name	administrator
Role	Admin

7. Click **Save**.

15.5.2. Configuring PKI authentication by using the roxctl CLI

You can configure PKI authentication by using the **roxctl** CLI.

Procedure

- Run the following command:

```
$ roxctl -e <hostname>:<port_number> central userpki create -c <ca_certificate_file> -r
<default_role_name> <provider_name>
```

15.5.3. Updating authentication keys and certificates

You can update your authentication keys and certificates by using the RHACS portal.

Procedure

1. Create a new authentication provider.

2. Copy the role mappings from your old authentication provider to the new authentication provider.

3. Rename or delete the old authentication provider with the old root CA key.

15.5.4. Logging in by using a client certificate

After you configure PKI authentication, users see a certificate prompt on the RHACS portal login page. The prompt only shows up if a client certificate trusted by the configured root CA is installed on the user's system.

Use the procedure described in this section to log in by using a client certificate.

Procedure

1. Open the RHACS portal.

2. Select a certificate in the browser prompt.

3. On the login page, select the authentication provider name option to log in with a certificate. If you do not want to log in by using the certificate, you can also log in by using the administrator password or another login method.
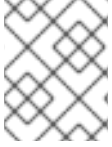
> **NOTE**
>
> Once you use a client certificate to log into the RHACS portal, you cannot log in with a different certificate unless you restart your browser.

CHAPTER 16. USING THE SYSTEM HEALTH DASHBOARD

The Red Hat Advanced Cluster Security for Kubernetes system health dashboard provides a single interface for viewing health related information about Red Hat Advanced Cluster Security for Kubernetes components.

> **NOTE**
>
> The system health dashboard is only available on Red Hat Advanced Cluster Security for Kubernetes 3.0.53 and newer.

16.1. SYSTEM HEALTH DASHBOARD DETAILS

To access the health dashboard:

- On the RHACS portal, navigate to **Platform Configuration → System Health**.

The health dashboard organizes information in the following groups:

- **Cluster Health** – Shows the overall state of Red Hat Advanced Cluster Security for Kubernetes cluster.

- **Vulnerability Definitions** – Shows the last update time of vulnerability definitions.

- **Image Integrations** – Shows the health of all registries that you have integrated.

- **Notifier Integrations** – Shows the health of any notifiers (Slack, email, Jira, or other similar integrations) that you have integrated.

- **Backup Integrations** – Shows the health of any backup providers that you have integrated.

The dashboard lists the following states for different components:

- **Healthy** – The component is functional.

- **Degraded** – The component is partially unhealthy. This state means the cluster is functional, but some components are unhealthy and require attention.

- **Unhealthy** – This component is not healthy and requires immediate attention.

- **Uninitialized** – The component has not yet reported back to Central to have its health assessed. An uninitialized state may sometimes require attention, but often components report back the health status after a few minutes or when the integration is used.

Cluster health section

The **Cluster Overview** shows information about your Red Hat Advanced Cluster Security for Kubernetes cluster health. It reports the health information about the following:

- **Collector Status** – It shows whether the Collector pod that Red Hat Advanced Cluster Security for Kubernetes uses is reporting healthy.

- **Sensor Status** – It shows whether the Sensor pod that Red Hat Advanced Cluster Security for Kubernetes uses is reporting healthy.

- **Sensor Upgrade** – It shows whether the Sensor is running the correct version when compared with Central.

- **Credential Expiration** - It shows if the credentials for Red Hat Advanced Cluster Security for Kubernetes are nearing expiration.

> **NOTE**
>
> Clusters in the **Uninitialized** state are not reported in the number of clusters secured by Red Hat Advanced Cluster Security for Kubernetes until they check in.

Vulnerabilities definition section

The **Vulnerabilities Definition** section shows the last time vulnerability definitions were updated and if the definitions are up to date.

Integrations section

There are 3 integration sections **Image Integrations**, **Notifier Integrations**, and **Backup Integrations**. Similar to the **Cluster Health** section, these sections list the number of unhealthy integrations if they exist. Otherwise, all integrations report as healthy.

> **NOTE**
>
> The **Integrations** section lists the healthy integrations as **0** if any of the following conditions are met:
>
> - You have not integrated Red Hat Advanced Cluster Security for Kubernetes with any third-party tools.
>
> - You have integrated with some tools, but disabled the integrations, or have not set up any policy violations.

16.2. GENERATING A DIAGNOSTIC BUNDLE BY USING THE RHACS PORTAL

You can generate a diagnostic bundle by using the system health dashboard on the RHACS portal.

Prerequisites

- To generate a diagnostic bundle, you need **read** permission for the **DebugLogs** resource.

Procedure

1. On the RHACS portal, select **Platform Configuration → System Health**.

2. On the **System Health** view header, click **Generate Diagnostic Bundle**.

3. For the **Filter by clusters** drop-down menu, select the clusters for which you want to generate the diagnostic data.

4. For **Filter by starting time**, specify the date and time (in UTC format) from which you want to include the diagnostic data.

5. Click **Download Diagnostic Bundle**.

16.2.1. Additional resources

- Generating a diagnostic bundle