



# Red Hat Advanced Cluster Security for Kubernetes 3.74

**roxctl CLI**

roxctl CLI



roxctl CLI

## Legal Notice

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

## Abstract

This document describes how to install and use the `roxctl` command-line interface, including the `roxctl` syntax and operations. It provides some common command examples.

## Table of Contents

<b>CHAPTER 1. GETTING STARTED WITH THE ROXCTL CLI .....</b>	<b>3</b>
1.1. INSTALLING THE ROXCTL CLI	3
1.1.1. Installing the roxctl CLI by downloading the binary	3
1.1.1.1. Installing the roxctl CLI on Linux	3
1.1.1.2. Installing the roxctl CLI on macOS	3
1.1.1.3. Installing the roxctl CLI on Windows	4
1.1.2. Running the roxctl CLI from a container	4
1.2. AUTHENTICATING USING THE ROXCTL CLI	5
1.3. USING THE ROXCTL CLI	6
1.3.1. Managing Central's database	6
Backing up Central database	6
Restoring Central database	6
1.3.2. Managing secured clusters	7
Generating Sensor deployment files	7
Installing Sensor by using the generate YAML files	7
Downloading Sensor bundle for existing clusters	8
Deleting cluster integration	8
1.3.3. Checking policy compliance	8
Configuring output format	8
Checking deployment YAML files	9
Checking images	9
Checking image scan results	9
1.3.4. Debugging issues	10
Managing Central log level	10
Viewing the logs	10
Viewing current log level	10
Changing the log level	10
Retrieving debugging information	10
1.3.5. Generating build-time network policies	10



# CHAPTER 1. GETTING STARTED WITH THE ROXCTL CLI

**roxctl** is a command-line interface (CLI) for running commands on Red Hat Advanced Cluster Security for Kubernetes. This topic describes **roxctl** syntax, operations, and provides some common examples.

## 1.1. INSTALLING THE ROXCTL CLI

You can install the **roxctl** CLI by downloading the binary or you can run the **roxctl** CLI from a container image.

### 1.1.1. Installing the roxctl CLI by downloading the binary

You can install the **roxctl** CLI to interact with Red Hat Advanced Cluster Security for Kubernetes from a command-line interface. You can install **roxctl** on Linux, Windows, or macOS.

#### 1.1.1.1. Installing the roxctl CLI on Linux

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

##### Procedure

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

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.74.4/bin/Linux/roxctl
```

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

```
$ chmod +x roxctl
```

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

```
$ echo $PATH
```

##### Verification

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

```
$ roxctl version
```

#### 1.1.1.2. Installing the roxctl CLI on macOS

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

##### Procedure

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

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.74.4/bin/Darwin/roxctl
```

2. Remove all extended attributes from the binary:

```
$ xattr -c roxctl
```

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

```
$ chmod +x roxctl
```

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

```
$ echo $PATH
```

### Verification

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

```
$ roxctl version
```

### 1.1.1.3. Installing the roxctl CLI on Windows

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

#### Procedure

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

```
$ curl -O https://mirror.openshift.com/pub/rhacs/assets/3.74.4/bin/Windows/roxctl.exe
```

### Verification

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

```
$ roxctl version
```

### 1.1.2. Running the roxctl CLI from a container

The **roxctl** client is the default entry point in Red Hat Advanced Cluster Security for Kubernetes **roxctl** image. To run the **roxctl** client in a container image:

#### Prerequisites

- You must first generate an authentication token from the RHACS portal.

#### Procedure

1. Log in to the **registry.redhat.io** registry.

```
$ docker login registry.redhat.io
```

2. Pull the latest container image for the **roxctl** CLI.

```
$ docker pull registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:3.74.4
```



■

After you install the CLI, you can run it by using the following command:

```
$ docker run -e ROX_API_TOKEN=$ROX_API_TOKEN \
-it registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:3.74.4 \
-e $ROX_CENTRAL_ADDRESS <command>
```

## Verification

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

```
$ docker run -it registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:3.74.4 version
```

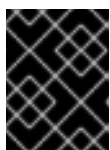
## 1.2. AUTHENTICATING USING THE ROXCTL CLI

For authentication, you can use an authentication token or your administrator password. Red Hat recommends using an authentication token in a production environment because each token is assigned specific access control permissions.

Use the following steps to generate an authentication token.

### Procedure

1. Navigate to the RHACS portal.
2. Go to **Platform Configuration → Integrations**.
3. Scroll down to the **Authentication Tokens** category, and click **API Token**.
4. Click **Generate Token**.
5. Enter a name for the token and select a role that provides the required level of access (for example, **Continuous Integration** or **Sensor Creator**).
6. Click **Generate**.



### IMPORTANT

Copy the generated token and securely store it. You will not be able to view it again.

**NOTE**

After you have generated the authentication token, export it as **ROX\_API\_TOKEN** variable:

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

You can also save the token in a file and use it with the **--token-file** option. For example:

```
$ roxctl central debug dump --token-file <token_file>
```

- You cannot use both the **-password (-p)** and the **--token-file** options simultaneously.
- If you have already set **ROX\_API\_TOKEN** variable, and specify the **--token-file** option, the **roxctl** CLI uses the specified token file for authentication.
- If you have already set **ROX\_API\_TOKEN** variable, and specify the **--password** option, the **roxctl** CLI uses the specified password for authentication.

## 1.3. USING THE ROXCTL CLI

Review the following sections to learn how to complete common tasks using the CLI.

**NOTE**

- Export the following variables before using these commands:

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

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

- You can use the **--help** option to get more information about the commands.

### 1.3.1. Managing Central's database

Central stores information about:

- Activity observed in your clusters,
- Information retrieved from integrated image registries or scanners, and
- Red Hat Advanced Cluster Security for Kubernetes configuration.

You can back up and restore Central's database by using the **roxctl** CLI.

#### Backing up Central database

Run the following command to back up Central's database:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" central backup
```

#### Restoring Central database

Run the following command to restore Central's database:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" central db restore <backup_filename>
```

### 1.3.2. Managing secured clusters

To secure a Kubernetes or an OpenShift Container Platform cluster, you must deploy Red Hat Advanced Cluster Security for Kubernetes services into the cluster. You can generate deployment files in the RHACS portal by navigating to the **Platform Configuration → Clusters** view, or you can use the **roxctl** CLI.

#### Generating Sensor deployment files

##### Kubernetes

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" sensor generate k8s --name <cluster_name> --central "$ROX_CENTRAL_ADDRESS"
```

##### OpenShift Container Platform

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" sensor generate openshift --openshift-version <ocp-version> --name <cluster_name> --central "$ROX_CENTRAL_ADDRESS" 1
```

- 1** For the **--openshift-version** option specify the major OpenShift Container Platform version number for your cluster. For example, specify **3** for OpenShift Container Platform version **3.x** and specify **4** for OpenShift Container Platform version **4.x**.

Read the **--help** output to see other options that you might need to use depending on your system architecture.

Verify that the endpoint you provide for **--central** can be reached from the cluster where you are deploying Red Hat Advanced Cluster Security for Kubernetes services.

#### NOTE

If you are using a non-gRPC capable load balancer, such as HAProxy, AWS Application Load Balancer (ALB), or AWS Elastic Load Balancing (ELB):

- Use the WebSocket Secure (**wss**) protocol. To use **wss**, prefix the address with **wss://**, and
- Add the port number after the address, for example:

```
$ roxctl sensor generate k8s --central wss://stackrox-central.example.com:443
```

#### Installing Sensor by using the generate YAML files

When you generate the Sensor deployment files, **roxctl** creates a directory called **sensor-<cluster\_name>** in your working directory. The script to install Sensor is present in this directory. Run the sensor installation script to install Sensor.

```
$ ./sensor-<cluster_name>/sensor.sh
```

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

### Downloading Sensor bundle for existing clusters

Use the following command to download Sensor bundles for existing clusters by specifying a cluster name or ID.

```
$ roxctl sensor get-bundle <cluster_name_or_id>
```

### Deleting cluster integration

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" cluster delete --name=<cluster_name>
```



#### IMPORTANT

Deleting cluster integration will not remove Red Hat Advanced Cluster Security for Kubernetes services running in the cluster. You can remove them by running the **delete-sensor.sh** script from the Sensor installation bundle.

### 1.3.3. Checking policy compliance

You can use the **roxctl** CLI to check deployment YAML files and images for policy compliance.

#### Configuring output format

When you check policy compliance by using the **deployment check**, **image check**, or **image scan** commands, you can specify the output format by using the **-o** option. This option determines how the output of a command is displayed in the terminal.

You can change the output format by adding the **-o** option to the command and specifying the format as **json**, **table**, **csv**, or **junit**.

For example, the following command checks a deployment and then displays the result in **csv** format:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" \
  deployment check --file =<yaml_filename> \
  -o csv
```



#### NOTE

When you do not specify the **-o** option for the output format, the following default behavior is used:

- The format for the **deployment check** and the **image check** commands is **table**.
- The default output format for the **image scan** command is **json**. This is the old JSON format output for compatibility with older versions of the CLI. To get the output in the new JSON format, specify the option with format, as **-o json**.

Different options are available to configure the output. The following table lists the options and the format in which they are available.

Option	Description	Formats
<b>--compact-output</b>	Use this option to display the JSON output in a compact format.	<b>json</b>

Option	Description	Formats
<b>--headers</b>	Use this option to specify custom headers.	<b>table</b> and <b>csv</b>
<b>--no-header</b>	Use this option to omit the header row from the output.	<b>table</b> and <b>csv</b>
<b>--row-jsonpath-expressions</b>	Use this option to specify <a href="#">GJSON paths</a> to select specific items from the output. For example, to get the <b>Policy name</b> and <b>Severity</b> for a deployment check, use the following command:  <pre>\$ roxctl -e "\$ROX_CENTRAL_ADDRESS" \   deployment check --file=&lt;yaml_filename&gt; \   -o table --headers POLICY-NAME,SEVERITY \   --row-jsonpath-expressions="{results.#.violatedPolicies.#.name,results.#.violatedPolicies.#.severity}"</pre>	<b>table</b> and <b>csv</b>
<b>--merge-output</b>	Use this options to merge table cells that have the same value.	<b>table</b>
<b>headers-as-comment</b>	Use this option to include the header row as a comment in the output.	<b>csv</b>
<b>--junit-suite-name</b>	Use this option to specify the name of the JUnit test suite.	<b>junit</b>

### Checking deployment YAML files

The following command checks build-time and deploy-time violations of your security policies in YAML deployment files. Use this command to validate:

- Configuration options in a YAML file, such as resource limits or privilege options; or
- Aspects of the images used in a YAML file, such as components or vulnerabilities.

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" deployment check --file=<yaml_filename>
```

### Checking images

The following command checks build-time violations of your security policies in images.

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" image check --image=<image_name>
```

### Checking image scan results

You can also check the scan results for specific images.

The following command returns the components and vulnerabilities found in the image in JSON format. The format is defined in the API reference.

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" image scan --image <image_name>
```

To cause Red Hat Advanced Cluster Security for Kubernetes to re-pull image metadata and image scan results from the associated registry and scanner, add the **--force** option.



#### NOTE

To check specific image scan results, you must have a token with both **read** and **write** permissions for the **Image** resource. The default **Continuous Integration** system role already has the required permissions.

### 1.3.4. Debugging issues

#### Managing Central log level

Central saves information to its container logs.

#### Viewing the logs

You can see the container logs for Central by running:

##### Kubernetes

```
$ kubectl logs -n stackrox <central_pod>
```

##### OpenShift Container Platform

```
$ oc logs -n stackrox <central_pod>
```

#### Viewing current log level

You can change the log level to see more or less information in Central logs. Run the following command to view the current log level:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" central debug log
```

#### Changing the log level

Run the following command to change the log level:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" central debug log --level=<log_level> 1
```

**1** The acceptable values for **<log\_level>** are **Panic**, **Fatal**, **Error**, **Warn**, **Info**, and **Debug**.

#### Retrieving debugging information

To gather debugging information for investigating issues, run the following command:

```
$ roxctl -e "$ROX_CENTRAL_ADDRESS" central debug dump
```

### 1.3.5. 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>)? 1
```

**1** 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> 1
```

**1** Specify the path of the folder that has the Kubernetes manifests.

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

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

### Additional resources

- [Using build-time network policy generator](#)