Red Hat Advanced Cluster Security for Kubernetes 4.1

roxctl CLI

roxctl CLI
roxctl CLI
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

**CHAPTER 1. GETTING STARTED WITH THE ROXCTL CLI**  ........................................... 3

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 BY USING THE ROXCTL CLI ......................................... 5
   1.2.1. Creating an API token ................................................................. 5
   1.2.2. Exporting and saving the authentication token .............................. 6
   1.2.3. Configuring an authentication provider in Central ......................... 6

1.3. USING THE ROXCTL CLI ............................................................................ 7
   1.3.1. Managing Central’s database ....................................................... 8
       Backing up Central database ............................................................... 8
       Restoring Central database ............................................................... 8
   1.3.2. Managing secured clusters .......................................................... 8
       Generating Sensor deployment files ................................................. 8
       Installing Sensor by using the generate YAML files .......................... 9
       Downloading Sensor bundle for existing clusters ............................. 9
       Deleting cluster integration .............................................................. 9
   1.3.3. Checking policy compliance .......................................................... 10
       Configuring output format ............................................................... 10
       Checking deployment YAML files .................................................... 11
       Checking images ............................................................................... 11
       Checking image scan results ............................................................ 11
   1.3.4. Debugging issues ............................................................................ 12
       Managing Central log level ............................................................... 12
       Viewing the logs ............................................................................... 12
       Viewing current log level ............................................................... 12
       Changing the log level ..................................................................... 12
       Retrieving debugging information .................................................... 12
   1.3.5. Generating build-time network policies ....................................... 12
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/4.1.2/bin/Linux/roxctl
   ```

2. Make the roxctl binary executable:

   ```
   $ chmod +x roxctl
   ```

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

   ```
   $ echo $PATH
   ```

Verification

- Verify the roxctl version you have installed:

  ```
  $ roxctl version
  ```

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/4.1.2/bin/Darwin/roxctl
   ```

2. Remove all extended attributes from the binary:
3. Make the `roxctl` binary executable:

   ```bash
   $ chmod +x roxctl
   ```

4. Place the `roxctl` binary in a directory that is on your `PATH`:

   To check your `PATH`, execute the following command:

   ```bash
   $ echo $PATH
   ```

Verification

- Verify the `roxctl` version you have installed:

  ```bash
  $ roxctl version
  ```

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:

  ```bash
  $ curl -O https://mirror.openshift.com/pub/rhacs/assets/4.1.2/bin/Windows/roxctl.exe
  ```

Verification

- Verify the `roxctl` version you have installed:

  ```bash
  $ 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.

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

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

   ```bash
   $ docker pull registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:4.1.2
   ```
After you install the CLI, you can run it by using the following command:

```bash
$ docker run -e ROX_API_TOKEN=$ROX_API_TOKEN \
   -it registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:4.1.2 \
   -e $ROX_CENTRAL_ADDRESS <command>
```

**NOTE**

In Red Hat Advanced Cluster Security Cloud Service, when using `roxctl` commands that require the Central address, use the **Central instance address** as displayed in the **Instance Details** section of the Red Hat Hybrid Cloud Console. For example, use `acs-ABCD12345.acs.rhcloud.com` instead of `acs-data-ABCD12345.acs.rhcloud.com`.

**Verification**

- Verify the `roxctl` version you have installed.

```bash
$ docker run -it registry.redhat.io/advanced-cluster-security/rhacs-roxctl-rhel8:4.1.2 version
```

## 1.2. AUTHENTICATING BY USING THE `ROXCTL` CLI

For authentication, you can use an API token, your administrator password, or `roxctl login` command.

- Use an API token in a production environment with continuous integration (CI). Each token is assigned specific access permissions, providing control over the actions it can perform. In addition, API tokens do not require interactive processes, such as browser-based logins, making them ideal for automated processes. These tokens have a time-to-live (TTL) of 1 year, providing a longer validity period for seamless integration and operational efficiency.

- Use your administrator password only for testing purposes and not in the production environment.

- Use the `roxctl login` command only for interactive use.

### 1.2.1. Creating an API token

**Procedure**

1. In the RHACS portal, navigate to **Platform Configuration → Integrations**.

2. Scroll to the **Authentication Tokens** category, and then click **API Token**.

3. Click **Generate Token**.

4. Enter a name for the token and select a role that provides the required level of access (for example, **Continuous Integration** or **Sensor Creator**).

5. Click **Generate**.
IMPORTANT
Copy the generated token and securely store it. You will not be able to view it again.

1.2.2. Exporting and saving the authentication token

Procedure

1. After you have generated the authentication token, export it as the `ROX_API_TOKEN` variable by entering the following command:

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

2. (Optional): You can also save the token in a file and use it with the `--token-file` option by entering the following command:

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

Note the following guidelines:

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

1.2.3. Configuring an authentication provider in Central

You can configure an authentication provider in Central and initiate the login process with the `roxctl` CLI. Set the `ROX_ENDPOINT` variable, initiate the login process with the `roxctl central login` command, select the authentication provider in a browser window, and retrieve the token information from the `roxctl` CLI as described in the following procedure.

Prerequisite

1. You selected an authentication provider of your choice, such as OpenID Connect (OIDC) with fragment or query mode.

Procedure

1. Run the following command to set the `ROX_ENDPOINT` variable to Central hostname and port:

   ```bash
   export ROX_ENDPOINT=<central_hostname:port>
   ```

2. Run the following command to initiate the login process to Central:

   ```bash
   $ roxctl central login
   ```

3. Within the `roxctl` CLI, a URL is printed as output and you are redirected to a browser window where you can select the authentication provider you want to use.
4. Log in with your authentication provider.
   After you have successfully logged in, the browser window indicates that authentication was 
   successful and you can close the browser window.

5. The roxctl CLI displays your token information including details such as the access token, the 
   expiration time of the access token, the refresh token if one has been issued, and notification 
   that these values are stored locally.

Example output

Please complete the authorization flow in the browser with an auth provider of your choice. If no 
browser window opens, please click on the following URL:

http://127.0.0.1:xxxxx/login

INFO: Received the following after the authorization flow from Central:
INFO: Access token: <redacted> 1
INFO: Refresh token: <redacted> 3
INFO: Storing these values under $HOME/.roxctl/login… 4

1. The access token.
2. The expiration time of the access token.
3. The refresh token.
4. The directory where values of the access token, the access token expiration time, and the 
   refresh token are stored locally.

IMPORTANT

Ensure that you set the environment to determine the directory where the 
configuration is stored. By default, the configuration is stored in the

$HOME/.roxctl/roxctl-config directory.

- If you set the $ROX_CONFIG_DIR environment variable, the configuration is 
  stored in the $ROX_CONFIG_DIR/roxctl-config directory. This option has 
  the highest priority.

- If you set the $XDG_RUNTIME_DIR environment variable and the 
  $ROX_CONFIG_DIR variable is not set, the configuration is stored in the 
  $XDG_RUNTIME_DIR/roxctl-config directory.

- If you do not set the $ROX_CONFIG_DIR or $XDG_RUNTIME_DIR 
  environment variable, the configuration is stored in the 
  $HOME/.roxctl/roxctl-config directory.

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.

- In Red Hat Advanced Cluster Security Cloud Service, when using `roxctl` commands that require the Central address, use the Central instance address as displayed in the Instance Details section of the Red Hat Hybrid Cloud Console. For example, use `acs-ABCD12345.acs.rhcloud.com` instead of `acs-data-ABCD12345.acs.rhcloud.com`.

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

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:

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

**Restoring Central database**

Run the following command to restore Central’s database:

```bash
$ 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**

```bash
$ 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"

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.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td>--compact-output</td>
<td>Use this option to display the JSON output in a compact format.</td>
<td>json</td>
</tr>
<tr>
<td>--headers</td>
<td>Use this option to specify custom headers.</td>
<td>table and csv</td>
</tr>
<tr>
<td>--no-header</td>
<td>Use this option to omit the header row from the output.</td>
<td>table and csv</td>
</tr>
</tbody>
</table>
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.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
<th>Formats</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>--row-jsonpath-expressions</code></td>
<td>Use this option to specify JSON paths to select specific items from the output. For example, to get the <strong>Policy name</strong> and <strong>Severity</strong> for a deployment check, use the following command:</td>
<td><strong>table</strong> and <strong>csv</strong></td>
</tr>
</tbody>
</table>
|                                             | `$ roxctl -e "$ROX_CENTRAL_ADDRESS" \                              
|                                             |   deployment check --file=<yaml_filename> \                              
|                                             |   -o table --headers POLICY-NAME,SEVERITY \                              
|                                             |   --row-jsonpath-expressions="" \                                      
|                                             |   {results.#.violatedPolicies.#.name,results.#.violatedPolicies.#.severity}" |               |
| `--merge-output`                           | Use this options to merge table cells that have the same value.            | **table**     |
| `headers-as-comment`                       | Use this option to include the header row as a comment in the output.      | **csv**       |
| `--junit-suite-name`                       | Use this option to specify the name of the JUnit test suite.               | **junit**     |
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>

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>)?
   ```

   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>
   ```

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

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

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
-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.

Additional resources

- Using build-time network policy generator