Red Hat build of Cryostat 2

Using the Red Hat build of Cryostat Operator to configure Cryostat

Last Updated: 2023-12-12

Legal Notice

Abstract

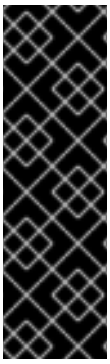
Red Hat build of Cryostat is a Red Hat offering on OpenShift Container Platform. Use the Using the Red Hat build of Cryostat Operator to configure Cryostat to learn how to use the Red Hat build of Cryostat Operator to configure Cryostat.

Table of Contents

PREFACE

The Red Hat build of Cryostat is a container-native implementation of JDK Flight Recorder (JFR) that you can use to securely monitor the Java Virtual Machine (JVM) performance in workloads that run on an OpenShift Container Platform cluster. You can use Cryostat 2.4 to start, stop, retrieve, archive, import, and export JFR data for JVMs inside your containerized applications by using a web console or an HTTP API.

Depending on your use case, you can store and analyze your recordings directly on your Red Hat OpenShift cluster by using the built-in tools that Cryostat provides or you can export recordings to an external monitoring application to perform a more in-depth analysis of your recorded data.



IMPORTANT

Red Hat build of Cryostat 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 .

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message.

CHAPTER 1. RED HAT BUILD OF CRYOSTAT OPERATOR

You can use the Red Hat build of Cryostat Operator to manage and configure your Cryostat instance. The Red Hat build of Cryostat Operator is available on the OpenShift Container Platform (OCP).

1.1. OVERVIEW OF THE RED HAT BUILD OF CRYOSTAT OPERATOR

After you create or update a Cryostat application on the OpenShift Container Platform, the Red Hat build of Cryostat Operator creates and manages the Cryostat application.

Operator level 2 seamless upgrades

From Cryostat 2.2, the Operator Capability Level for the Red Hat build of Cryostat Operator is set to **Level 2 Seamless Upgrades** on the Operator Lifecycle Manager framework. After you upgrade your Red Hat build of Cryostat Operator, the Red Hat build of Cryostat Operator automatically upgrades Cryostat and its related components. The automatic upgrade operation does not remove any JFR recordings, templates, rules, and other stored components, from your Cryostat instance.

NOTE

The automatic upgrade operation occurs only for minor releases or patch update releases of Cryostat. For major releases, you might need to re-install the Red Hat build of Cryostat Operator.

Persistent volume claims

You can create persistent volume claims (PVCs) on Red Hat OpenShift with the Red Hat build of Cryostat Operator so that your Cryostat application can store archived recordings on a cloud storage disk.

Operator configuration settings

Additionally, you can make the following changes to the default configuration settings for the Red Hat build of Cryostat Operator:

- Configure the PVC that was created by the Red Hat build of Cryostat Operator, so that your Cryostat application can store archived recordings on a cloud storage disk.
- Configure your Cryostat application to trust TLS certificates from specific applications.
- Deploy Cryostat as a minimal deployment, so that the operator requires less resources to deploy a Cryostat application.
- Disable cert-manager, so that the operator does not need to generate self-signed certificates for Cryostat components.
- Install custom event template files, which are located in ConfigMaps, to your Cryostat instance, so you can use the templates to create recordings when Cryostat starts.

From Cryostat 2.2, the following configuration options for the Red Hat build of Cryostat Operator are included:

- Resource requirements, which you can use to specify resource requests or limits for the **core**, **datasource**, or **grafana** containers.
- Service customization, so that you can control the services that the Red Hat build of Cryostat Operator creates.

- Sidecar report options, which the Red Hat build of Cryostat Operator can use to provision one or more report generators for your Cryostat application.

Single-namespace or multi-namespace Cryostat instances

The Red Hat build of Cryostat Operator provides both a **Cryostat** API and a **Cluster Cryostat** API. You can use the **Cryostat** API to create Cryostat instances that work in a single namespace. You can use the **Cluster Cryostat** API to create Cryostat instances that work across multiple namespaces. You can control these Cryostat instances by using a GUI that is accessible from the Red Hat OpenShift web console.

Users who can access the multi-namespace Cryostat instance have access to all target applications in any namespace that is visible to that Cryostat instance. Therefore, when you deploy a multi-namespace Cryostat instance, you must consider which namespaces to select for monitoring, which namespace to install Cryostat into, and which users can have access rights.

Prerequisites for configuring the Red Hat build of Cryostat Operator

Before you configure the Red Hat build of Cryostat Operator, ensure that the following prerequisites are met:

- Installed the Red Hat build of Cryostat Operator in a project on Red Hat OpenShift.
- Created a Cryostat instance by using the Red Hat build of Cryostat Operator.

Additional resources

- See Operator Capability Levels (Operator SDK)
- See Installing Cryostat on Red Hat OpenShift using an operator (Installing Cryostat)

1.2. EXCLUDING SUPPORTIVE CONTAINERS

You can choose to exclude supportive applications from getting deployed with your Cryostat application. Supportive applications are the supportive containers that are listed with your Cryostat pod. When you exclude supportive containers, fewer system resources are required to deploy your Cryostat application.

By default, Cryostat sets the **minimal** property in your project's Red Hat build of Cryostat Operator YAML configuration file to **false**. With this configuration, the Red Hat build of Cryostat Operator deploys your Cryostat application with all the standard supportive applications, such as **jfr-datasource** and the Grafana dashboard, which are contained in the same pod as your Cryostat application. These supportive applications can interact with your Cryostat data and provide you with additional capabilities for interacting with this data.

The Red Hat build of Cryostat Operator defaults to the following configurations:

- Deploys a pre-configured Grafana application.
- Deploys a **jfr-datasource** application for converting JDK Flight Recorder (JFR) data to JSON, which is a readable format for Grafana.
- Includes a Dashboard JSON file that is pre-configured in Grafana when you deploy Cryostat.

You can set the **minimal** property to **true**, so that the Red Hat build of Cryostat Operator automatically restarts your Cryostat instance as a minimal deployment. This means that the operator deploys only those applications that are listed in your Cryostat container and ignores any standard supportive

applications, such as **jfr-datasource** and the Grafana dashboard, that are contained in the same pod as your Cryostat application.

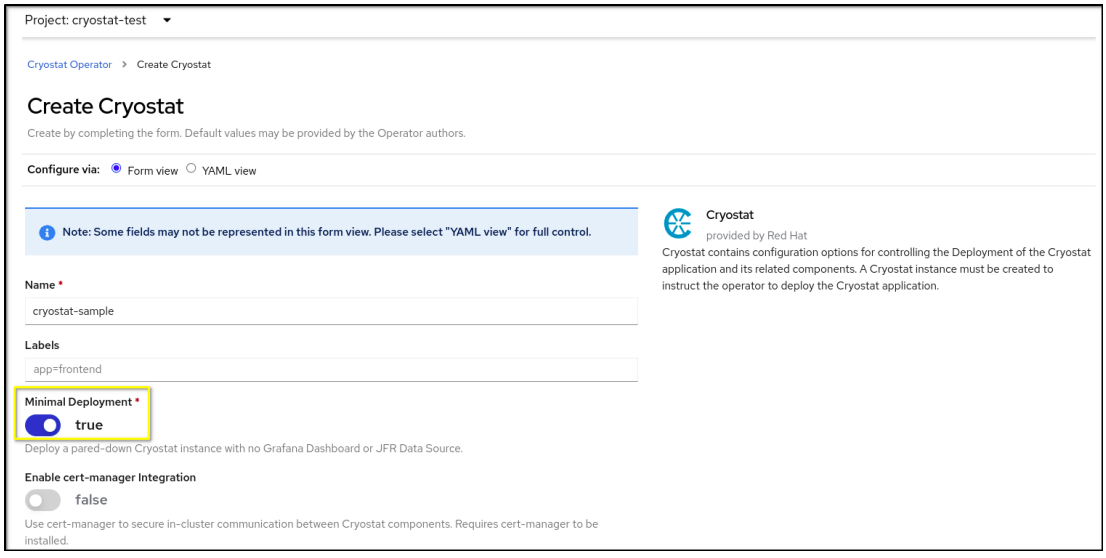
Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.

Procedure

1. On your Red Hat OpenShift web console, click **Operators > Installed Operators**.
2. From the list of available operators, select Red Hat build of Cryostat.
3. Click the **Details** tab.
4. In the **Provided APIs** section, the **Cryostat** and **Cluster Cryostat** custom resources (CRs) are available. Select one of the following options:
   - To create a single-namespace Cryostat instance, select Cryostat, then click **Create instance**.
   - To create a multi-namespace instance of Cryostat, select **Cluster Cryostat**, then click **Create instance**.
5. To configure the **minimal** property, choose one of the following options:
   a. Click the **Form view** radio button.
      i. Set the **Minimal Deployment** switch to **true**. You must also enter a value in the **Name** field.

Figure 1.1. Toggling the Minimal Deployment switch to true



      ii. Click **Create**. Depending on the type of instance you created, the instance opens under one of the following tabs:
         - If you created a single-namespace Cryostat instance, the instance is available under the **Cryostat** tab on the **Operator details** page.
         - If you created a Cluster Cryostat instance, the instance is available under the **Cluster Cryostat** tab on the **Operator details** page.

b. Click the **YAML view** radio button.

i. Change the value for the **minimal** property to **true** in the **spec:** key set.

Example of configuring the minimal property

```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  minimal: true
--
```

ii. Click **Save**.

Verification

1. From the Red Hat OpenShift web console, select the project where you created your Cryostat instance or the project you chose as the **Install Namespace** of your Cluster Cryostat instance.
2. Navigate to **Workloads → Deployments**.
3. From the list of deployments, select the deployment that matches the name of your Cryostat or Cluster Cryostat instance. A **Deployment details** page opens on your web console.
4. Navigate to the **Containers** section. A single listed container indicates that the Red Hat build of Cryostat Operator has deployed your Cryostat application as a minimal deployment.

Additional resources

- For more information about the OpenShift CLI, see Getting started with the OpenShift CLI (Red Hat OpenShift documentation)
- See Creating a JDK Flight Recorder (JFR) recording (Creating a JFR recording with Cryostat)

1.3. DISABLING CERT-MANAGER

You can disable cert-manager functionality by configuring the **enableCertManager** property of the Red Hat build of Cryostat Operator.

By default, Red Hat build of Cryostat Operator's **enableCertManager** property is set to **true**. This means that the Red Hat build of Cryostat Operator uses the cert-manager **CA** issuer to generate self-signed certificates for your Cryostat components. The Red Hat build of Cryostat Operator uses these certificates to enable HTTPS communication among Cryostat components operating in a cluster.

You can set the **enableCertManager** property to **false**, so that the Red Hat build of Cryostat Operator does not need to generate self-signed certificates for Cryostat components.

IMPORTANT

If you set the **enableCertManager** property to **false**, you could introduce potential security implications from unencrypted internal traffic to the cluster that contains your running Cryostat application.

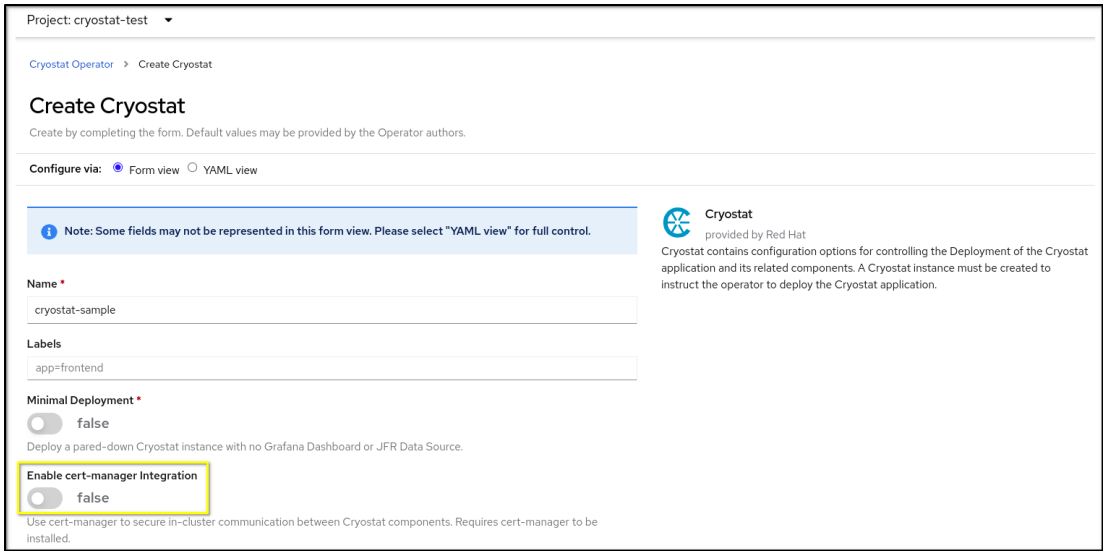
Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.

Procedure

1. Navigate to **Operators > Installed Operators** on your OpenShift web console.
2. From the list of available operators, select Red Hat build of Cryostat.
3. Click the **Details** tab.
4. In the **Provided APIs** section, the **Cryostat** and **Cluster Cryostat** custom resources (CRs) are available. Select one of the following options:
   - To create a single-namespace Cryostat instance, select Cryostat, then click **Create instance**.
   - To create a multi-namespace instance of Cryostat, select **Cluster Cryostat**, then click **Create instance**.
5. To configure the **enableCertManager** property, choose one of the following options:
   a. Click the **Form view** radio button.
      i. Set the **Enable cert-manager Integration** switch to **false**, and then enter a value in the **Name** field.

      Figure 1.2. Toggling the Enable cert-manager Integration switch to false

      ii. Click **Create**. Depending on the type of instance you created, the instance opens under one of the following tabs:
         - If you created a single-namespace Cryostat instance, the instance is available under the **Cryostat** tab on the **Operator details** page.
         - If you created a Cluster Cryostat instance, the instance is available under the **Cluster Cryostat** tab on the **Operator details** page.

   b. Click the **YAML view** radio button.

i. In the **spec:** key set of the YAML file, change the **enableCertManager** property to **false**.

Example of configuring the spec: key set in a YAML file

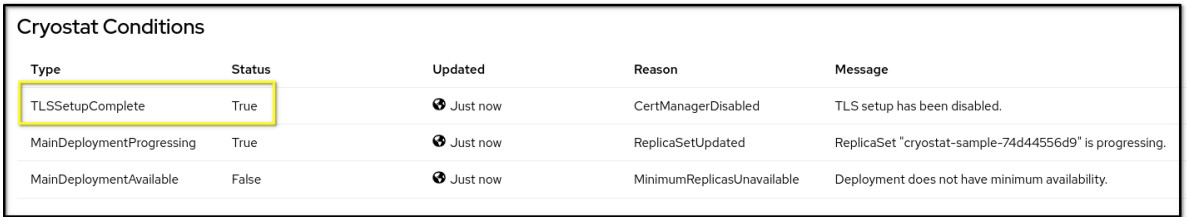
```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  enableCertManager: false
--
```

ii. Click the **Save** button.
The Red Hat build of Cryostat Operator automatically restarts your Cryostat application, enabling the application to run with the updated **enableCertManager** property configuration.

Verification

1. Select your Cryostat or Cluster Cryostat instance:
   - If you created a Cryostat instance, select your Cryostat instance from the **Cryostat** tab on the **Operator details** page.
   - If you created a Cluster Cryostat instance, select your Cluster Cryostat instance from the **Cluster Cryostat** tab on the **Operator details** page.
2. Navigate to the **Cryostat Conditions** table.
3. Verify that the **TLSSetupComplete** condition is set to **true** and that the **Reason** column for this condition is set to **CertManagerDisabled**. This indicates that you have set the **enableCertManager** property to **false**.

Figure 1.3. Example showing the TLSSetupComplete condition set to true



Additional resources

- See the cert-manager documentation
- See Creating a JDK Flight Recorder (JFR) recording (Creating a JFR recording with Cryostat)

1.4. CUSTOMIZING EVENT TEMPLATES

In Cryostat 2, you can configure the **eventTemplates** property of the Red Hat build of Cryostat Operator YAML configuration file to include multiple custom templates. An event template outlines the event recording criteria for your JDK Flight Recording (JFR). You can configure a JFR through its

associated event template.

By default, Red Hat build of Cryostat Operator includes some pre-configured event templates. These pre-configured event templates might not meet your needs, so you can use Red Hat build of Cryostat Operator to generate custom event templates for your Cryostat instance and store these templates in ConfigMaps for easier retrieval. You can generate a custom event template in the following ways:

- Use the Red Hat OpenShift web console to upload an event template into a custom resource.
- Edit the YAML file for your Cryostat custom resource on the Red Hat OpenShift web console.

After you store a custom event template in a **ConfigMap**, you can deploy a new Cryostat instance with this custom event template. You can then use your custom event template with JFR to monitor your Java application to meet your needs.

Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.
- Logged in to your Cryostat web console.

Procedure

1. To download a default event template, navigate to your Cryostat web console and from the **Events** menu, click **Downloads**.

   NOTE

   Event templates are in XML format and have a file name extension of **.jfc**.

2. *Optional:* If you want a custom event template, edit the downloaded default event template by using a text editor or XML editor to configure the template to meet your needs.
3. Log in to your Red Hat OpenShift web console by entering the **oc login** command in your CLI.
4. Create a **ConfigMap** resource from the event template by entering the following command in your CLI. You must issue the command in the path where you want to deploy your Cryostat application. You can use this resource to store an event template file that is inside the cluster where you run your Cryostat instance.

   Example of creating a ConfigMap resource by using the CLI

   ```
   $ oc create configmap <template_name> --from-file=<path_to_custom_event_template>
   ```

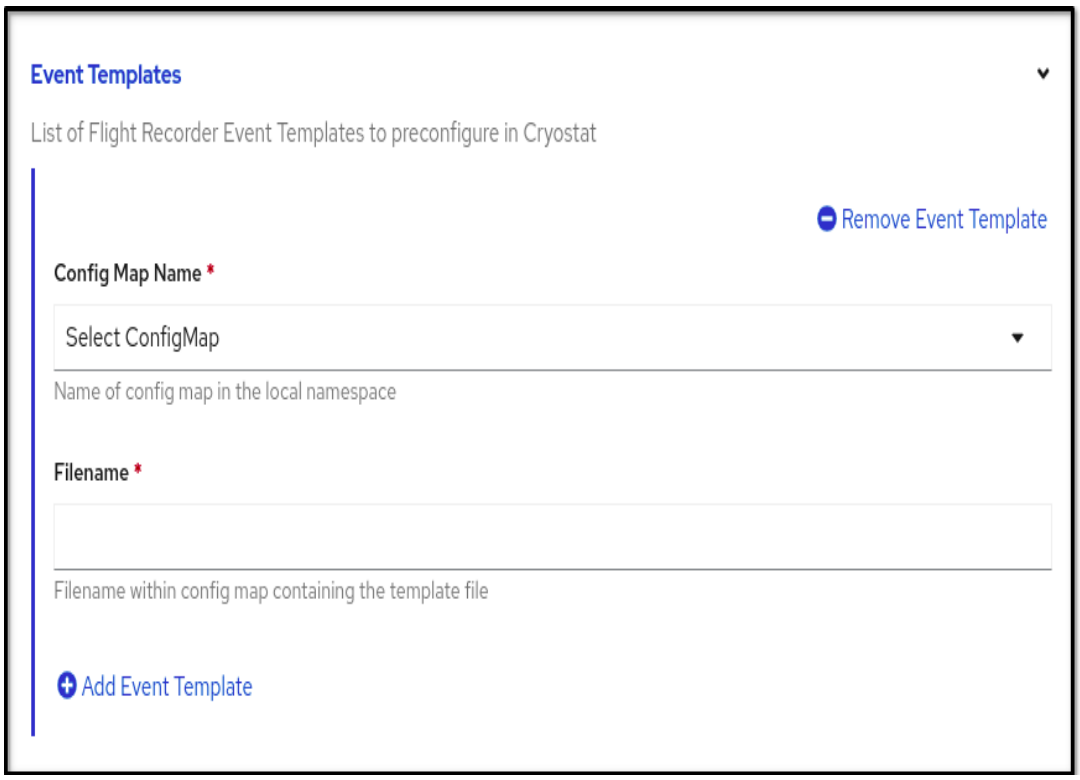
5. On your Red Hat OpenShift web console, click **Operators > Installed Operators**.
6. From the list of available operators, select Red Hat build of Cryostat.
7. Under the **Details** tab on the **Operator details** page, create a Cryostat or Cluster Cryostat instance.

   a. In the **Provided APIs** section, the **Cryostat** and **Cluster Cryostat** custom resources (CRs) are available. Select one of the following options:

- To create a single-namespace Cryostat instance, select Cryostat, then click **Create instance**.
- To create a multi-namespace instance of Cryostat, select **Cluster Cryostat**, then click **Create instance**.

8. Choose one of the following options to upload an event template in XML format into a resource:
   a. Click the **Form view** radio button.
      i. Navigate to the **Event Templates** section of the Cryostat or Cluster Cryostat instance.
      ii. From the **Event Templates** menu, click **Add Event Template**. An **Event Templates** section opens on your Red Hat OpenShift console.
      iii. From the **Config Map Name** drop-down list, select the ConfigMap resource that contains your event template.

      Figure 1.4. Event Templates option for a Cryostat instance

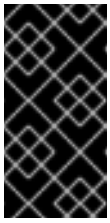


      iv. In the **Filename** field, enter the name of the **.jfc** file that is contained within your ConfigMap.
      v. To generate a Cryostat or a Cluster Cryostat instance with your custom event template, click **Create**.

   b. Click the **YAML view** radio button.
      i. Specify any custom event templates for the **eventTemplates** property. This property points the Red Hat build of Cryostat Operator to your ConfigMap, so that the Red Hat build of Cryostat Operator can read the event template.

      Example of specifying custom event templates for the eventTemplates property

```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  eventTemplates:
  - configMapName: custom-template1
    filename: my-template1.jfc
  - configMapName: custom-template2
    filename: my-template2.jfc
--
```



IMPORTANT

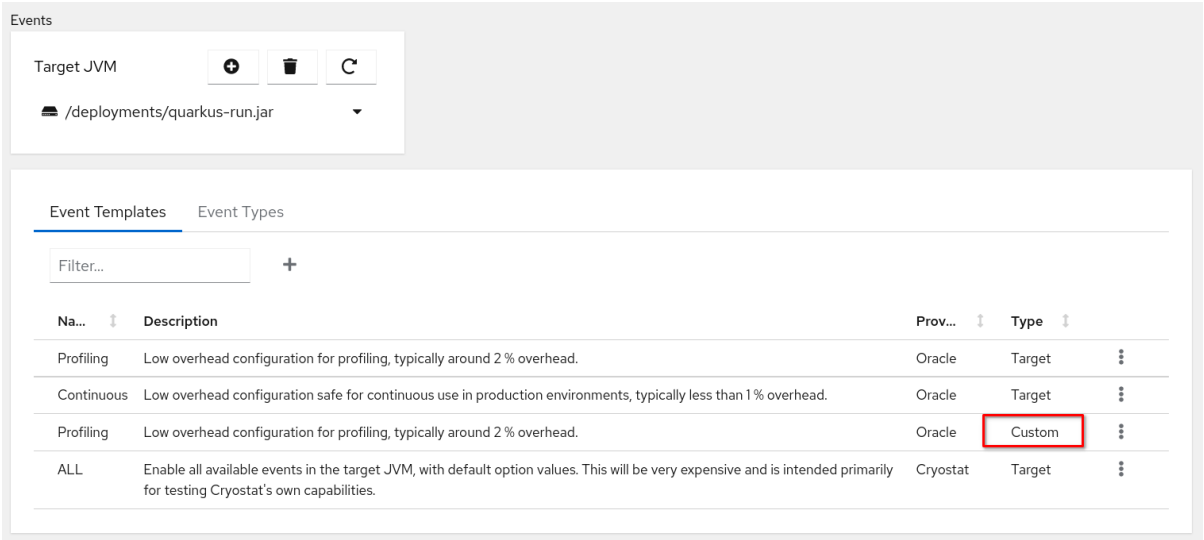
You must select the name of a ConfigMap, which is associated with your Cryostat or Cluster Cryostat instance, from the **configMapName** drop-down list. Additionally, you must specify a key associated with the ConfigMap in the **filename** field.

The Red Hat build of Cryostat Operator can now provide the custom event template as an XML file to your Cryostat application. Your custom event template opens alongside default event templates in your Cryostat web console.

Verification

1. On the Cryostat web console, click **Events** from the menu. If an **Authentication Required** window opens on your web console, enter your credentials and click **Save**.
2. Under the **Event Templates** tab, check if your custom event template shows in the list of available event templates.

Figure 1.5. Example of a listed custom event template under the Event Templates tab



Additional resources

- See Installing Cryostat on OpenShift using an operator (Installing Cryostat)
- See Accessing Cryostat by using the web console (Installing Cryostat)

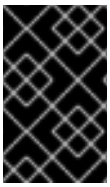
- See Using custom event templates (Using Cryostat to manage a JFR recording)

1.5. CONFIGURING TLS CERTIFICATES

You can specify the Red Hat build of Cryostat Operator to configure Cryostat to trust TLS certificates from specific applications.

Cryostat attempts to open a JMX connection to a target JVM that uses a TLS certificate. For a successful JMX connection, the Cryostat must pass all its authentication checks on the target JVM certificate.

You can specify multiple TLS secrets in the **trustedCertSecrets** array of the Red Hat build of Cryostat Operator YAML configuration file. You must specify the secret located in the same namespace as your Cryostat application in the **secretName** property of the array. The **certificateKey** property defaults to **tls.crt**, but you can change the value to an X.509 certificate file name.



IMPORTANT

Configuring a TLS certificate is required only for applications that have enabled TLS for remote JMX connections by using the **com.sun.management.jmxremote.registry.ssl=true** attribute.

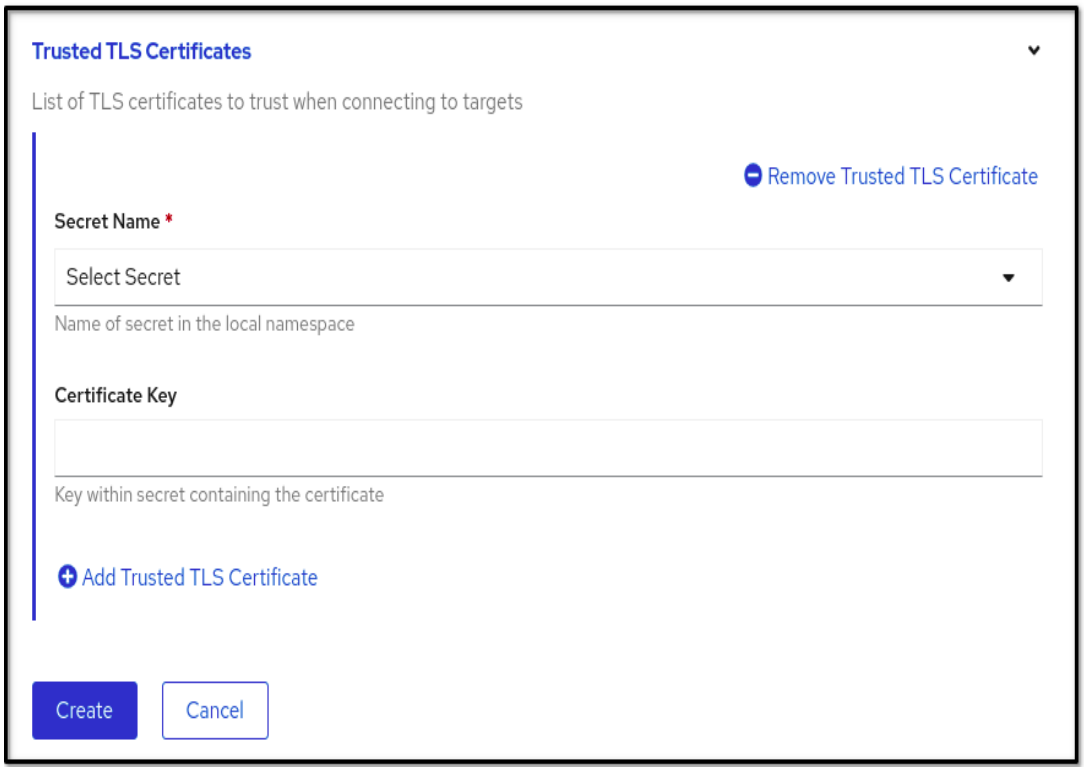
Prerequisites

- Logged in to the OpenShift Container Platform by using the OpenShift web console.
- Logged in to your Cryostat web console.

Procedure

1. On your Red Hat OpenShift web console, click **Operators > Installed Operators**.
2. From the list of available operators, select Red Hat build of Cryostat.
3. On the **Operator details** page, click the **Details** tab.
4. In the **Provided APIs** section, the **Cryostat** and **Cluster Cryostat** custom resources (CRs) are available. Select one of the following options:
    a. To create a single-namespace Cryostat instance, select Cryostat, then click **Create instance**.
    b. To create a multi-namespace instance of Cryostat, select **Cluster Cryostat**, then click **Create instance**.
5. To configure a TLS certificate, choose one of the following options:
    a. Click the **Form view** radio button.
        i. In the **Name** field, specify a name for the instance of Cryostat that you want to create.
        ii. Expand the **Trusted TLS Certificates** option, then click **Add Trusted TLS Certificate**. A list of options displays on your Red Hat OpenShift web console.

Figure 1.6. The Trusted TLS Certificates option



iii. Select a TLS secret from the **Secret Name** list. The **Certificate Key** field is optional.

NOTE

You can remove a TLS certificate by clicking **Remove Trusted TLS Certificate**.

iv. Click **Create**. Depending on the type of instance you created, the instance opens under one of the following tabs:

- If you created a single-namespace Cryostat instance, the instance is available under the **Cryostat** tab on the **Operator details** page.
- If you created a Cluster Cryostat instance, the instance is available under the **Cluster Cryostat** tab on the **Operator details** page.

b. Click the **YAML view** radio button.

i. Specify your secret, which is located in the same namespace as your Cryostat application, in the **secretName** property of the **trustedCertSecrets** array.

Example of specifying a secret in the trustedCertSecrets array

```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  trustedCertSecrets:
  - secretName: my-tls-secret
--
```

ii. *Optional:* Change the **certificateKey** property value to the application's X.509 certificate file name. If you do not change the value, the **certificateKey** property defaults to **tls.crt**.

Example of changing the certificateKey property's value

```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  trustedCertSecrets:
  - secretName: my-tls-secret
    certificateKey: ca.crt
--
```

iii. Click **Save**.
The Red Hat build of Cryostat Operator automatically restarts your Cryostat instance with the configured security settings.

Verification

1. Determine that all your application pods exist in the same OpenShift cluster namespace as your Cryostat pod by issuing the following command in your CLI:

```
$ oc get pods
```

2. Log in to the web console of your Cryostat instance.
3. On the **Dashboard** menu for your Cryostat instance, select a target JVM from the **Target** list.
4. In the navigation menu on the Cryostat web console, select **Recordings**. On the **Authentication Required dialog** window, enter your secret's credentials and then select **Save** to provide your credentials to the target JVM.

NOTE

If the selected target has password authentication enabled for JMX connections, you must provide the JMX credentials for the target JVM when prompted for a connection.

Cryostat connects to your application through the authenticated JMX connection. You can now use the **Recordings** and **Events** functions to monitor your application's JFR data.

Additional resources

- See Creating a JDK Flight Recorder (JFR) recording (Creating a JFR recording with Cryostat)
- See Installing Cryostat on Red Hat OpenShift using an operator (Installing Cryostat)
- See Accessing Cryostat by using the web console (Installing Cryostat)

1.6. CHANGING STORAGE VOLUME OPTIONS

You can use the Red Hat build of Cryostat Operator to configure storage volumes for your Cryostat or Cluster Cryostat instance. Cryostat supports persistent volume claim (PVC) and **emptyDir** storage volume types.

By default, Red Hat build of Cryostat Operator creates a PVC for your Cryostat or Cluster Cryostat instance that uses the default **StorageClass** resource with 500 mebibytes (MiB) of allocated storage.

You can create a custom PVC for your Cryostat application on OpenShift Container Platform by choosing one of the following options:

- Navigating to **Storage Options > PVC > Spec** in the **Form view** window, and then customizing your PVC by completing the relevant fields.
- Navigating to the **YAML view** window, and then editing the **storageOptions** array in the **spec: key** set to meet your needs.

NOTE

You can learn more about creating a custom PVC by navigating to Changing storage volume options in the *Using the Red Hat build of Cryostat Operator to configure Cryostat* guide.

You can configure the **emptyDir** storage volume for your Cryostat application on OpenShift Container Platform by choosing one of the following options:

- Enabling the **Empty Dir** setting in **Storage Options** on the **Form view** window.
- Setting the **spec.storageOptions.emptyDir.enabled** to **true** in the **YAML view** window.

Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.

Procedure

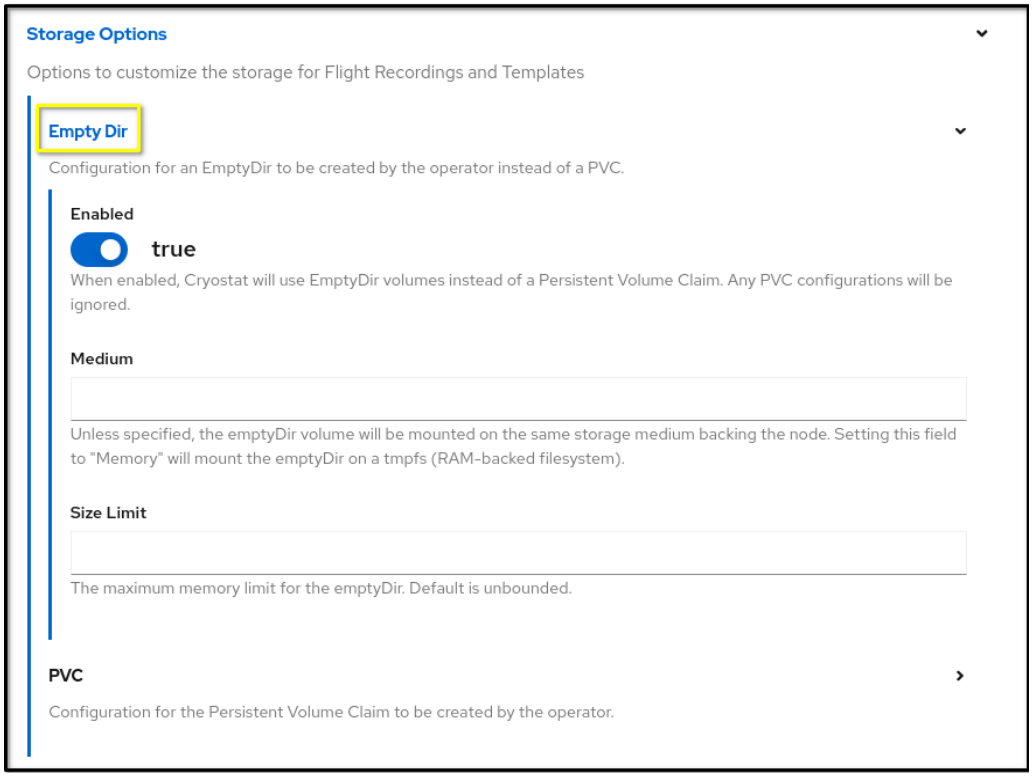
1. On your Red Hat OpenShift web console, click **Operators > Installed Operators**.
2. From the list of available operators, select Red Hat build of Cryostat.
   a. Click the **Details** tab.
3. In the **Provided APIs** section, the **Cryostat** and **Cluster Cryostat** custom resources (CRs) are available. Select one of the following options:
   - To create a single-namespace Cryostat instance, select Cryostat, then click **Create instance**.
   - To create a multi-namespace instance of Cryostat, select **Cluster Cryostat**, then click **Create instance**.
4. To change storage settings for your Cryostat application, choose one of the following options:
   a. Click the **Form view** radio button.

i. Navigate to the **Storage Options** section, and enter a value in the **Name** field.

ii. Expand **Storage Options** and click **Empty Dir**. An expanded selection of options opens on your Red Hat OpenShift web console.

iii. Set the **Enabled** switch to **true**.

Figure 1.7. Example showing the Empty Dir switch set to true



iv. Click **Create**. Depending on the type of instance you created, the instance opens under one of the following tabs:

- If you created a single-namespace Cryostat instance, the instance is available under the **Cryostat** tab on the **Operator details** page.
- If you created a Cluster Cryostat instance, the instance is available under the **Cluster Cryostat** tab on the **Operator details** page.

b. Click the **YAML view** radio button.

i. In the **spec:** key set of the YAML file, add the **storageOptions** definition and set the **emptyDir** property to **true**.

Example showing the emptyDir property set as true

```
--
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  storageOptions:
    emptyDir:
      enabled: true
```

```
      medium: "Memory"
      sizeLimit: 1Gi
--
```

ii. *Optional:* Set values for the **medium** and **sizeLimit** properties.

iii. Click the **Save** button. The Red Hat build of Cryostat Operator creates an **EmptyDir** volume for storage instead of creating a PVC for your Cryostat instance.

1.7. SCHEDULING OPTIONS FOR CRYOSTAT

From the Red Hat OpenShift web console, you can use the Red Hat build of Cryostat Operator to define policies for scheduling a Cryostat application and its generated reports to nodes.

You can define **Node Selector**, **Affinities**, and **Tolerations** definitions in the YAML configuration file for a Cryostat or Cluster Cryostat custom resource (CR) on Red Hat OpenShift. You must define these definitions under the **spec.SchedulingOptions** property for the Cryostat application and the **spec.ReportOptions.SchedulingOptions** property for the report generator sidecar. By specifying the **SchedulingOptions** property, the Cryostat application and its report generator sidecar pods will be scheduled on nodes that meet the scheduling criteria.

a targeted node application can receive sidecar reports updates from a Cryostat instance.

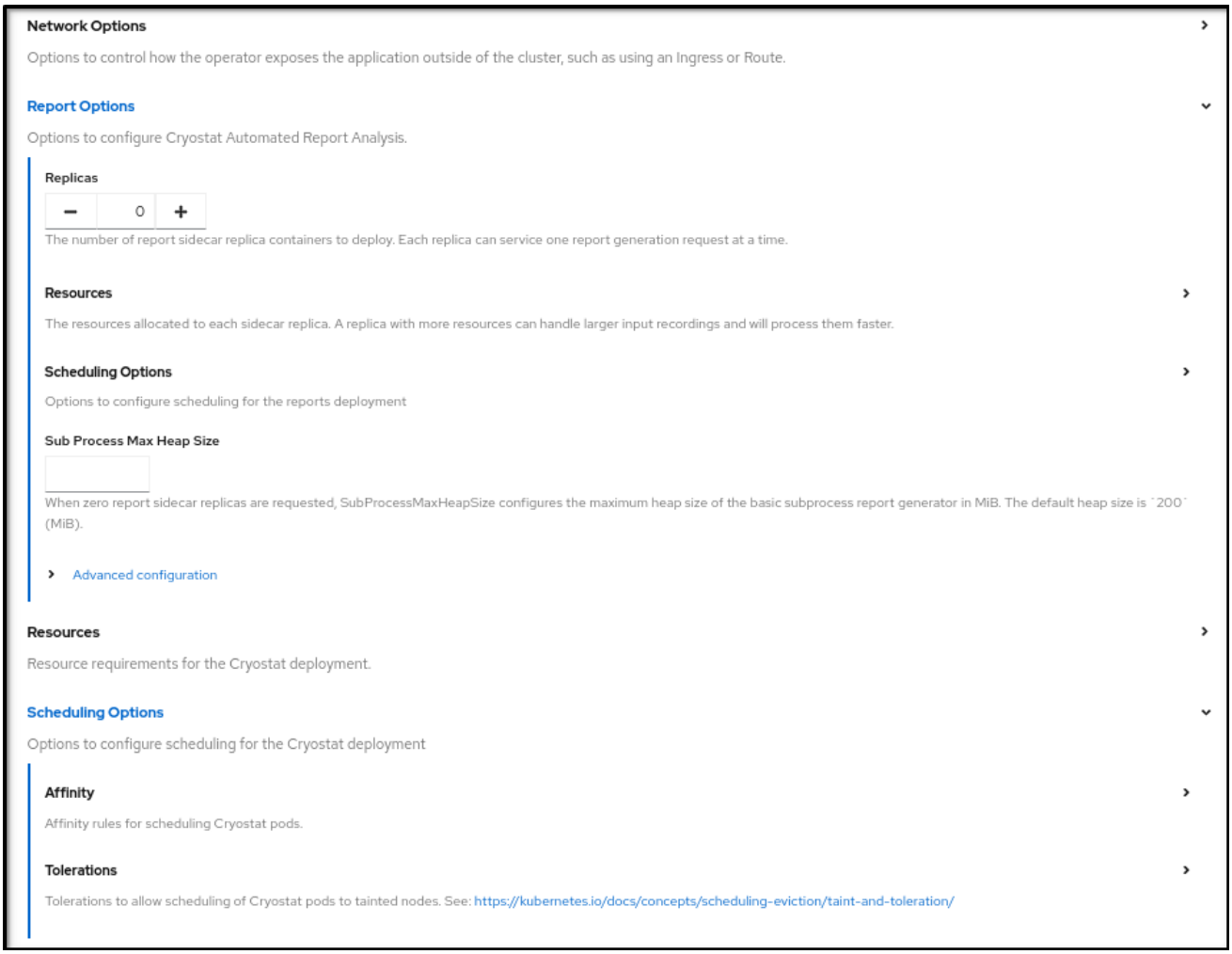
Example that shows the YAML configuration for a Cryostat CR that defines schedule options

```
kind: Cryostat
apiVersion: operator.cryostat.io/v1beta1
metadata:
  name: cryostat
spec:
  schedulingOptions:
    nodeSelector:
      node: good
    affinity:
      nodeAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
          nodeSelectorTerms:
          - matchExpressions:
            - key: node
              operator: In
              values:
              - good
              - better
      podAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchLabels:
              pod: good
          topologyKey: topology.kubernetes.io/zone
      podAntiAffinity:
        requiredDuringSchedulingIgnoredDuringExecution:
        - labelSelector:
            matchLabels:
              pod: bad
```

```
        topologyKey: topology.kubernetes.io/zone
    tolerations:
    - key: node
      operator: Equal
      value: ok
      effect: NoExecute
  reportOptions:
    replicas: 1
    schedulingOptions:
      nodeSelector:
        node: good
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: node
                operator: In
                values:
                - good
                - better
        podAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchLabels:
                pod: good
            topologyKey: topology.kubernetes.io/zone
        podAntiAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
          - labelSelector:
              matchLabels:
                pod: bad
            topologyKey: topology.kubernetes.io/zone
      tolerations:
      - key: node
        operator: Equal
        value: ok
        effect: NoExecute
```

Alternatively, you can open your Red Hat OpenShift web console, create a Cryostat instance, and then define **Affinities** and **Tolerations** definitions in the **SchedulingOptions** and **reportOptions.SchedulingOptions** options for that Cryostat instance.

Figure 1.8. The Report Options and Scheduling Options panels on the OpenShift web console



CHAPTER 2. POD SECURITY ADMISSION

Red Hat OpenShift uses Pod Security Admission (PSA) to apply a set of security rules for application pods that are in the same Red Hat OpenShift cluster. In the context of Cryostat, these application pods include a Cryostat pod and a Report sidecar pod. Optionally, you can enable the Report sidecar pod on a Cryostat custom resource (CR). If an application does not meet the policy standards, the application cannot run in your Red Hat OpenShift cluster.

Red Hat OpenShift 4.8 deprecates the **PodSecurityPolicy** API and uses the PSA instead. The PSA provides the following benefits:

- Includes a built-in controller that can enforce pod security standards for your application pods.
- Includes a set of pod security standards that define three different policies: **Privileged**, **Baseline**, and **Restricted**.

On Red Hat OpenShift, you can use the PSA with security context constraints (SCCs) to define policies for an Red Hat OpenShift cluster. By default, the **restricted-v2** SCC aligns with the **Restricted** pod security standard.

NOTE

By default, the security context for a Cryostat pod conforms to the **restricted-v2** SCC, which means that Red Hat OpenShift can admit the pod in namespaces that enforce the **Restricted** pod security standard.

The **Restricted** policy requires that the Red Hat build of Cryostat Operator configures the container security context as follows:

- Drops **ALL** capabilities
- Sets **allowPrivilegeEscaltion** to **false**

The **Restricted** policy requires that the Red Hat build of Cryostat Operator configures the pod security context as follows:

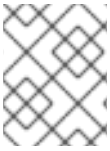
- Sets **runAsNonRoot** to **true**
- Sets the **seccompProfile** to **RuntimeDefault**

Additionally, the Red Hat build of Cryostat Operator defines **fsGroup** in the pod security context for the Cryostat application pod, so that Cryostat can read and write to files in a persistent storage volume on Red Hat OpenShift.

If you have additional requirements beyond conforming to the **Restricted** pod security standard, you can override the default security contexts that Cryostat uses.

2.1. CONFIGURING SECURITY CONTEXTS

You can specify pod and container security contexts in the Cryostat custom resource (CR) on Red Hat OpenShift. The security context applies permissions to the Cryostat pod, the Report sidecar pod (when it is in use), and the containers for each pod.



NOTE

If you change the settings of the CR, these settings override the default security context settings.

A security context applies specific permissions to an application that exists in a pod. The security context cannot change the criteria of an SCC policy. You can create a custom SCC to instruct the Red Hat OpenShift cluster to enforce strict permissions on the pod, such as actions that the pod can perform or resources that the pod can access.

To create a custom SCC you must have cluster administration permissions. You must also create a security context for any pods that operate in the cluster, so that these pods meet the custom SCC requirements.

An SCC enforces changes at the Red Hat OpenShift cluster level and namespace level, so that any pods operating inside this cluster receive policy criteria. By contrast, a security context is unique to a pod.

By default, the Red Hat build of Cryostat Operator conforms to the **restricted-v2** SCC policy for your Cryostat pod.

By default, the Red Hat build of Cryostat Operator creates a service account for Cryostat and its components, such as **jfr-datasource** and **grafana**.

To enable this service account to use a custom SCC, perform either of the following steps:

- Create a **Role Binding** that binds the Cryostat service account to a role that **uses** your custom SCC.
- Use a **Label Syncer** component to instruct your project's namespace to follow PSA policies.

NOTE

The **Label Syncer** component is outside the scope of this document. You cannot use the **Label Syncer** component on Red Hat OpenShift system namespaces, which are usually prefixed with the **openshift-** tag.

IMPORTANT

Before you configure a security context to apply specific permissions to an application pod, consider the security risks that you might introduce to your cluster on Red Hat OpenShift. The PSA provides three gradient policy levels that typically meet most requirements. Red Hat does not take any responsibility for security context changes that do not align with the Red Hat OpenShift pod security standards.

Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.
- Installed the Red Hat build of Cryostat Operator in a project on Red Hat OpenShift. See Installing Cryostat on Red Hat OpenShift by using a Red Hat build of Cryostat Operator (Installing Cryostat).
- *Optional:* Read the new PSA and new SCC policies. See Managing security context constraints (OpenShift Container Platform).

- *Optional:* Configured your project to use one of the three polices that the PSA provides.
    - If you want to use a custom SCC to enforce specific policies for your pod, you must configure the SCC to enable your pod's service account to access it.

Procedure

1. From the Red Hat OpenShift web console, click **Operators > Installed Operators**.
2. From the list of available operators, select Red Hat build of Cryostat.
3. Click **Provided APIs > Create** The Red Hat build of Cryostat Operator does not create a service account for the Report sidecar pods. Instead, these pods use default service accounts in their own namespaces.
4. To configure a security context, complete one of the following options:
    a. Click **YAML view**. From the **spec:** element, edit the **securityOptions** and **reportOptions** properties to match your security requirements.

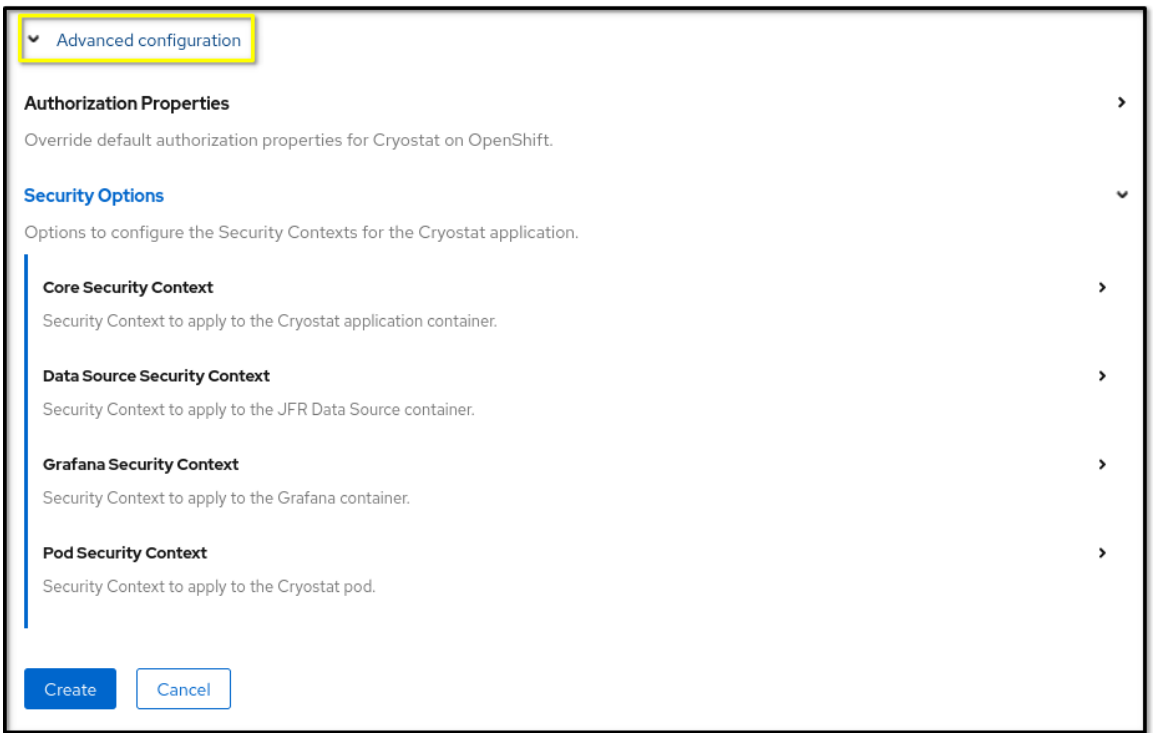
    Example configuration for a security context

```
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  securityOptions:
    podSecurityContext:
      runAsNonRoot: true
      seccompProfile:
        type: RuntimeDefault
    coreSecurityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop:
          - ALL
      runAsUser: 1001
    dataSourceSecurityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop:
          - ALL
    grafanaSecurityContext:
      allowPrivilegeEscalation: false
      capabilities:
        drop:
          - ALL
  reportOptions:
    replicas: 1
    podSecurityContext:
      runAsNonRoot: true
      seccompProfile:
        type: RuntimeDefault
    reportsSecurityContext:
      allowPrivilegeEscalation: false
```

```
capabilities:
  drop:
    - ALL
runAsUser: 1001
```

b. Expand **Advanced Configurations** to open additional options on your Red Hat OpenShift web console.

Figure 2.1. The Advanced configuration menu options



c. Expand **Core Security Context** From the available list of options, define settings for your security context.

5. Click **Create**.

6. Repeat step one through five for **Data Source Security Context**, **Grafana Security Context**, and **Pod Security Context** as appropriate.

7. *Optional:* If you are using the **Report Generator** service, you can also configure the security contexts for this service, as follows:

   a. From **Report Options**, expand **Advanced Configurations**.

   b. Expand **Security Options**. Define **Reports Security Context** and **Pod Security Context** as appropriate.

Additional resources

- Pod Security Standard policies .

2.2. POD SECURITY STANDARD POLICIES

The Pod Security Admission (PSA) includes three policies that cover security levels related to pod security standards. The following table explains each policy:

Profile	Description
Privileged	An unrestricted policy that provides a wide level of permissions for your Cryostat pod. Consider setting this policy if you need to provide known privilege escalations to your pods.
Baseline	Default policy that restricts known privileged escalations. The **Baseline** policy sets controls where each control defines restricted fields and allowed values.
Restricted	The **Restricted** policy that provides a low level of permissions for your Cryostat pod. This policy sets controls with each control defining restricted fields and allowed values.

CHAPTER 3. RBAC MAPPING CONFIGURATION

On OpenShift Container Platform (OCP), Cryostat uses a permission configuration that maps OCP resources to Cryostat-managed resources. The permission configuration provides Cryostat with a framework for authorizing a user to perform certain actions, such as creating a JFR recording, or viewing discovered targets.

The following table outlines definitions that represent Cryostat-managed resources:

Resource	Description
CERTIFICATE	SSL certificates that connect to Java Virtual Machine (JVM) applications with enabled encryption.
CREDENTIALS	Stored credentials for target JVM applications.
RECORDING	Recordings created for JVM applications.
REPORT	Report content generated from recordings.
RULE	Automated Rules that start recordings on matching targets when they become available to Cryostat, non-interactively.
TARGET	Discovered JVM applications to monitor.
TEMPLATE	Event templates to configure recordings.

The permission configuration defines lists of OCP resources that are equivalent to the previously listed resource definitions. API requests specify resource actions to translate a Cryostat-managed resource permissions into OCP resources. Cryostat checks each API request for this action and then processes the API request.

Cryostat assigns resource-verb pairs to each endpoint. These verbs are custom and specific to Cryostat. During permissions checks, Cryostat translates custom verbs into RBAC verbs.

You can implement the following verbs on these Cryostat-managed resources:

- **CREATE**: create
- **DELETE**: delete
- **READ**: get
- **UPDATE**: patch

The following example shows a mapping configuration that links a Cryostat-managed resource to a list of Red Hat OpenShift resources:

```
TARGET=pods,services
```

To create an API request that outputs a list of discovered JVM targets, for example, from the **Target JVM** pane on the **Recordings** page, you must have **READ** permissions to view the discoverable **TARGET**. In the RBAC system, the **READ** permission provides access to read pods and services.

By default, Cryostat uses the following RBAC mapping configuration.

```
auth.properties:
  TARGET=pods,services
  RECORDING=pods,pods/exec,cryostats.operator.cryostat.io
  CERTIFICATE=pods,cryostats.operator.cryostat.io
  CREDENTIALS=pods,cryostats.operator.cryostat.io
```

NOTE

The **ConfigMap** defines the mapping content. The previous example does not list all Cryostat-managed resources. If a Cryostat-managed resource is missing from the **ConfigMap**, Cryostat skips permission checks during the processing of an API request.

The Red Hat build of Cryostat Operator projects these settings from the provided **ConfigMap** API object into the Cryostat pod on Red Hat OpenShift. Your Cryostat pod can access these settings at any time to confirm what permissions of Cryostat functions a user can access. You can then define a **ClusterRole** in the custom resource (CR) that provides specific permissions to these mapped Red Hat OpenShift resources.

Example that shows a Cryostat CR with ConfigMap, ClusterRole, and filename fields defined in the spec field

```
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  authProperties:
    configMapName: auth-properties
    filename: auth.properties
    clusterRoleName: oauth-cluster-role
```

Additional resources

- See, RBAC permissions (Installing Cryostat).

3.1. CONFIGURING RBAC MAPPINGS

You can create a custom role with Cryostat-specific RBAC permissions and then bind this role to a user's Red Hat OpenShift account. This feature is useful for when you want to set specific permissions for each user that operates within the same Cryostat namespace.

Prerequisites

- Logged in to the OpenShift Container Platform by using the Red Hat OpenShift web console.
- Created a Cryostat instance in your project. See Installing Cryostat on Red Hat OpenShift using an operator (Installing Cryostat).

Procedure

1. Define a custom permission mapping in a **ConfigMap** object.

Example of a ConfigMap containing the permission mapping

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: auth-properties
data:
  auth.properties: |
    TARGET=pods,deployments.apps
    RECORDING=pods,pods/exec
    CERTIFICATE=deployments.apps,pods,cryostats.operator.cryostat.io
    CREDENTIALS=cryostats.operator.cryostat.io
```

To use custom permission mapping, a **ClusterRole** must exist and contain permissions for all Red Hat OpenShift objects listed in custom permission mapping.

Example of a ClusterRole that contains the necessary rules

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: additional-oauth-client
rules:
- apiGroups:
  - operator.cryostat.io
  resources:
  - cryostats
  verbs:
  - create
  - patch
  - delete
  - get
- apiGroups:
  - ""
  resources:
  - pods
  - pods/exec
  verbs:
  - create
  - patch
  - delete
  - get
- apiGroups:
  - apps
  resources:
  - deployments
  verbs:
  - create
  - patch
  - delete
  - get
```

After you enter your credentials on the Red Hat OpenShift web console, the **OAuth** server uses your credentials and the specified scope to generate an API token.

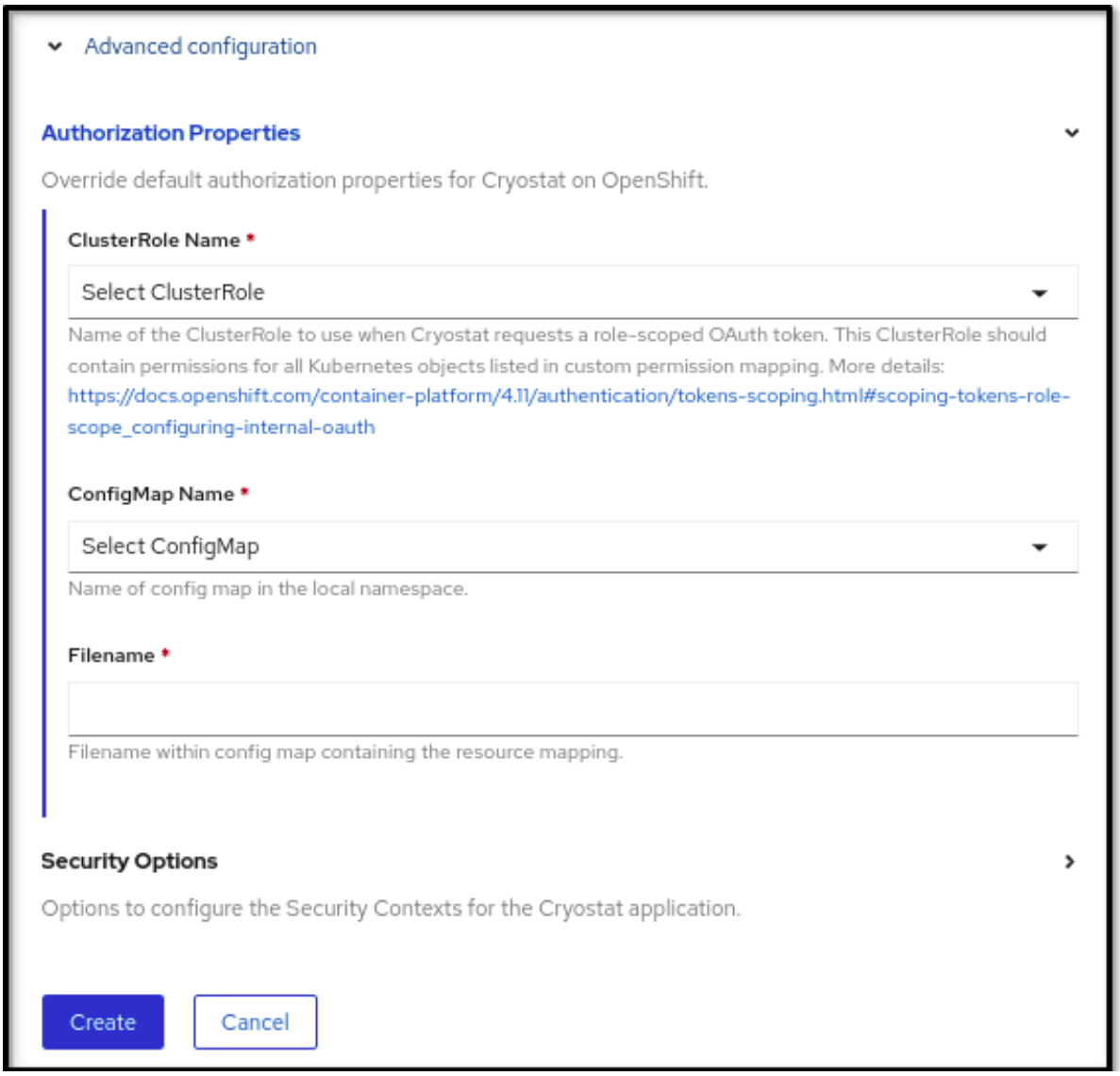
2. Provide the **authProperties** spec in the Cryostat Custom Resource (CR) to reference the **ConfigMap** that holds the mapping content, and **ClusterRole** that defines RBAC access for those mapped Red Hat OpenShift resources.

Example of a Cryostat CR with authProperties that define a custom permission mapping

```
apiVersion: operator.cryostat.io/v1beta1
kind: Cryostat
metadata:
  name: cryostat-sample
spec:
  authProperties:
 configMapName: auth-properties
 filename: auth.properties
 clusterRoleName: oauth-cluster-role
```

Alternatively, you can open your Red Hat OpenShift web console, create a Cryostat instance, and define **ClusterRole Name**, **ConfigMap Name**, and **Filename** properties in the **Authorization Properties** option, which you can access in the **Advanced configuration** section.

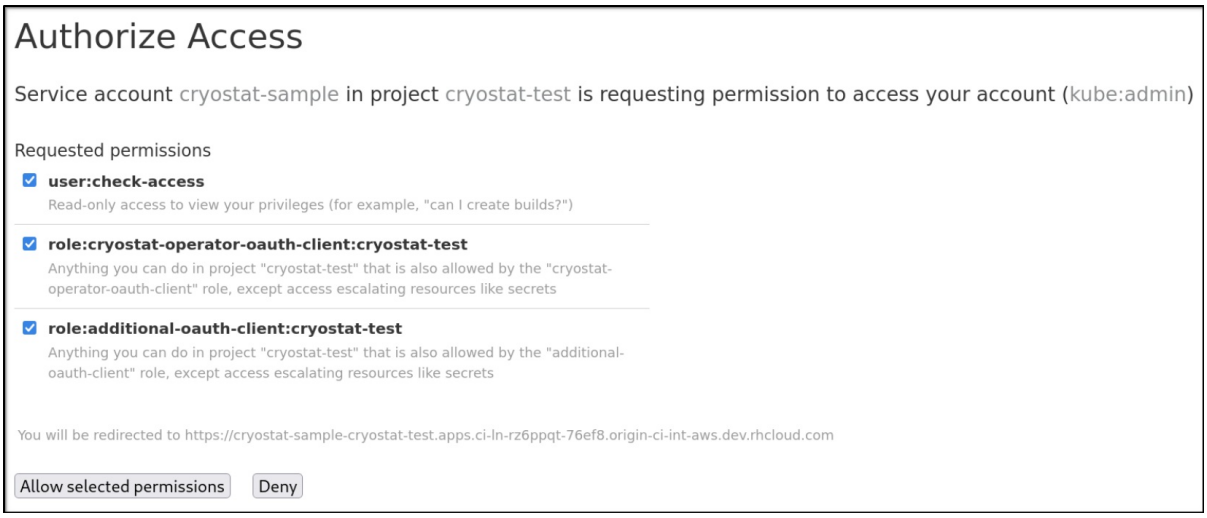
Figure 3.1. The Advanced configuration section on the OpenShift web console



Verification

1. From the **Installed Operators** menu, select your Cryostat instance.
2. Click the link in the **Application URL** section to access the login screen. The **OAuth** server redirects you to an OpenShift Container Platform login page.
3. Enter your credential details and then click **Login**. For the first time you log in through the **OAuth** server, an **Authorize Access** page opens on your web browser.
4. From the **Requested Permissions** option, confirm that the cluster role name matches the name that you specified in the Cryostat CR.
5. From the **Authorize Access** window, you can select the required checkboxes. For optimal Cryostat performance, select all checkboxes.

Figure 3.2. The Authorize Access window that lists three permissions



The **Authorize Access** window lists the following permissions:

- **User:check-access**, which is a permission check that the internal Cryostat application requests. Permission provides a user with read-only access to view their privileges.
- **role:cryostat-operator-oauth-client:<namespace>**, which is a permission check that the internal Cryostat application requests. Replace *<namespace>* with the name of your project name or your namespace from your CLI. Permission provides a user with access to complete any operations that the **cryostat-operator-oauth-client** role specifies, except access to escalate resources, such as secrets.
- **role:<user-define-clusterrole-name>:<namespace>**: The **clusterrole** that you defined in the Cryostat CR spec. Replace *<namespace>* with the name of your project name or your namespace from your CLI. Permission provides a user with access to complete any operations that the **additional-oauth-client role** specifies, except escalating access to resources, such as secrets.

6. Choose one of the following options:
   a. Click **Allow selected permissions** if you want to accept the selected requested permissions.
   b. Click **Deny** button if you want to reject all requested permission options.
   Your web browser redirects you to the Cryostat web console, where you can monitor Java applications that are running in a Java Virtual Machine (JVM).

Additional resources

- See, Installing Cryostat on Red Hat OpenShift by using a Red Hat build of Cryostat Operator (Installing Cryostat).

Revised on 2023-12-12 18:49:07 UTC