Red Hat Process Automation Manager 7.8

Deploying a Red Hat Process Automation Manager environment on Red Hat OpenShift Container Platform using Operators

Last Updated: 2022-04-07

Red Hat Process Automation Manager 7.8 Deploying a Red Hat Process Automation Manager environment on Red Hat OpenShift Container Platform using Operators

Red Hat Customer Content Services
brms-docs@redhat.com

Legal Notice

Abstract

This document describes how to deploy a Red Hat Process Automation Manager 7.8 environment on Red Hat OpenShift Container Platform using Operators.

Table of Contents

PREFACE

As a system engineer, you can deploy a Red Hat Process Automation Manager environment on Red Hat OpenShift Container Platform to provide an infrastructure to develop or execute processes and other business assets. You can use OpenShift Operators to deploy the environment defined in a structured YAML file and to maintain and modify this environment as necessary.

Prerequisites

- A Red Hat OpenShift Container Platform version 4 environment is available. For the exact versions of Red Hat OpenShift Container Platform that the current release supports, see Red Hat Process Automation Manager 7 Supported Configurations.
- At least four gigabytes of memory are available in the OpenShift environment.
- The OpenShift project for the deployment is created.
- You are logged in to the project using the OpenShift web console.
- Dynamic persistent volume (PV) provisioning is enabled. Alternatively, if dynamic PV provisioning is not enabled, enough persistent volumes must be available. By default, the deployed components require the following PV sizes:
  - Each KIE Server deployment by default requires one 1Gi PV for the database. You can change the database PV size. You can deploy multiple KIE Servers; each requires a separate database PV. This requirement does not apply if you use an external database server.
  - By default, Business Central requires one 1Gi PV. You can change the PV size for Business Central persistent storage.
  - Business Central Monitoring requires one 64Mi PV.
  - Smart Router requires one 64Mi PV.
- If you intend to scale any of the Business Central or Business Central Monitoring pods, your OpenShift environment supports persistent volumes with **ReadWriteMany** mode. If your environment does not support this mode, you can use NFS to provision the volumes. For information about access mode support in OpenShift public and dedicated clouds, see Access Modes.

CHAPTER 1. OVERVIEW OF RED HAT PROCESS AUTOMATION MANAGER ON RED HAT OPENSHIFT CONTAINER PLATFORM

You can deploy Red Hat Process Automation Manager into a Red Hat OpenShift Container Platform environment.

In this solution, components of Red Hat Process Automation Manager are deployed as separate OpenShift pods. You can scale each of the pods up and down individually to provide as few or as many containers as required for a particular component. You can use standard OpenShift methods to manage the pods and balance the load.

The following key components of Red Hat Process Automation Manager are available on OpenShift:

- KIE Server, also known as *Execution Server*, is the infrastructure element that runs decision services, process applications, and other deployable assets (collectively referred to as *services*) . All logic of the services runs on execution servers.
  A database server is normally required for KIE Server. You can provide a database server in another OpenShift pod or configure an execution server on OpenShift to use any other database server. Alternatively, KIE Server can use an H2 database; in this case, you cannot scale the pod.

  In some templates, you can scale up a KIE Server pod to provide as many copies as required, running on the same host or different hosts. As you scale a pod up or down, all of its copies use the same database server and run the same services. OpenShift provides load balancing and a request can be handled by any of the pods.

  You can deploy a separate KIE Server pod to run a different group of services. That pod can also be scaled up or down. You can have as many separate replicated KIE Server pods as required.

- Business Central is a web-based interactive environment used for authoring services. It also provides a management and monitoring console. You can use Business Central to develop services and deploy them to KIE Servers. You can also use Business Central to monitor the execution of processes.
  Business Central is a centralized application. However, you can configure it for high availability, where multiple pods run and share the same data.

  Business Central includes a Git repository that holds the source for the services that you develop on it. It also includes a built-in Maven repository. Depending on configuration, Business Central can place the compiled services (KJAR files) into the built-in Maven repository or (if configured) into an external Maven repository.

- Business Central Monitoring is a web-based management and monitoring console. It can manage the deployment of services to KIE Servers and provide monitoring information, but does not include authoring capabilities. You can use this component to manage staging and production environments.

- Smart Router is an optional layer between KIE Servers and other components that interact with them. When your environment includes many services running on different KIE Servers, Smart Router provides a single endpoint to all client applications. A client application can make a REST API call that requires any service. Smart Router automatically calls the KIE Server that can process a particular request.

You can arrange these and other components into various environment configurations within OpenShift.

CHAPTER 2. ARCHITECTURE OF AN AUTHORING ENVIRONMENT

In Red Hat Process Automation Manager, the Business Central component provides a web-based interactive user interface for authoring services. The KIE Server component runs the services.

The KIE Server uses a database server to store the state of process services.

You can also use Business Central to deploy services onto a KIE Server. You can use several KIE Servers to run different services and control the servers from the same Business Central.

Single authoring environment

In a single authoring environment, only one instance of Business Central is running. Multiple users can access its web interface at the same time, however the performance can be limited and there is no failover capability.

Business Central includes a built-in Maven repository that stores the built versions of the services that you develop (KJAR files/artifacts). You can use your continuous integration and continuous deployment (CICD) tools to retrieve these artifacts from the repository and move them as necessary.

Business Central saves the source code in a built-in Git repository, stored in the **.niogit** directory. It uses a built-in indexing mechanism to index the assets in your services.

Business Central uses persistent storage for the Maven repository and for the Git repository.

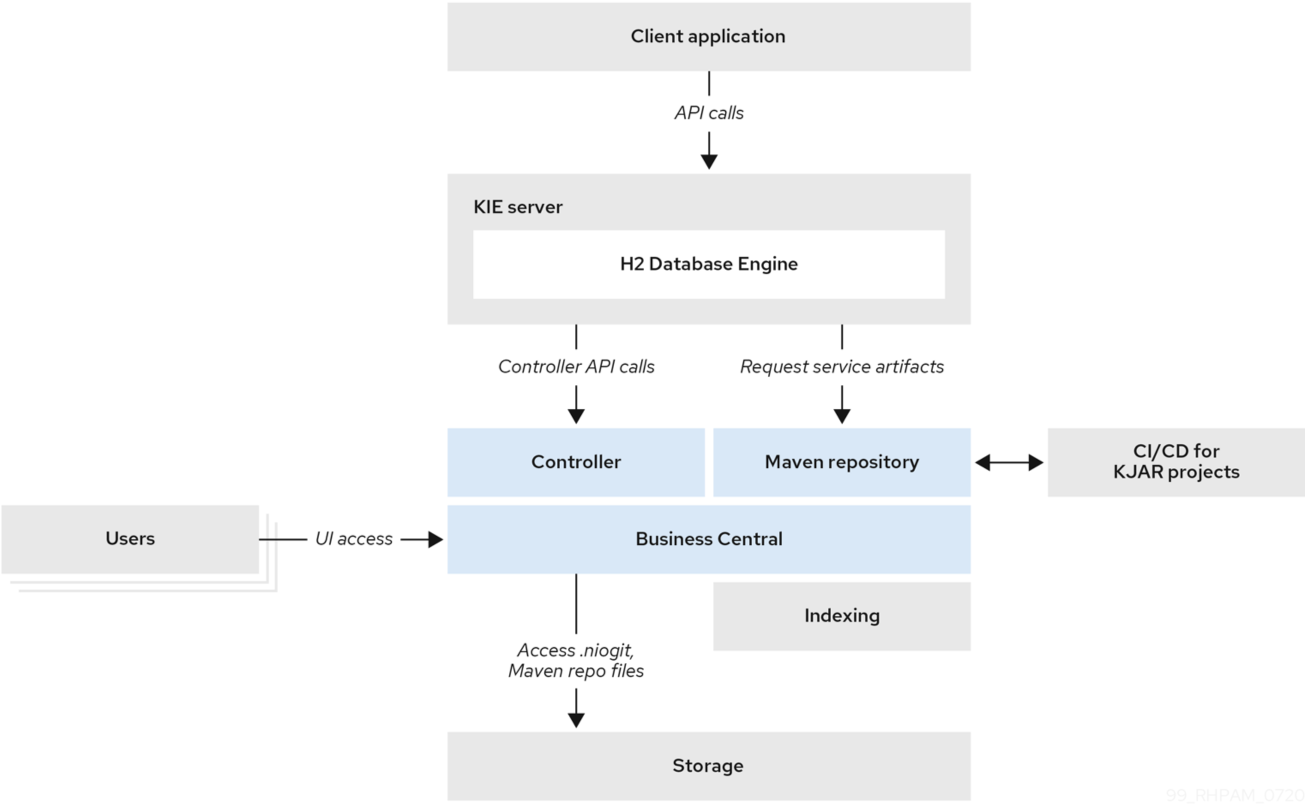
A single authoring environment, by default, includes one KIE Server. This KIE Server uses a built-in H2 database engine to store the state of process services.

A single authoring environment can use the *controller strategy*. Business Central includes the *Controller*, a component that can manage KIE Servers. When you configure a KIE Server to connect to Business Central, the KIE Server uses a REST API to connect to the Controller. This connection opens a persistent WebSocket. In an OpenShift deployment that uses the controller strategy, each KIE Server is initially configured to connect to the Business Central Controller.

When you use the Business Central user interface to deploy or manage a service on the KIE Server, the KIE Server receives the request through the Controller connection WebSocket. To deploy a service, the KIE Server requests the necessary artifact from the Maven repository that is a part of Business Central.

Client applications use a REST API to use services that run on the KIE Server.

Figure 2.1. Architecture diagram for a single authoring environment



Clustering KIE Servers and using multiple KIE Servers

You can scale a KIE Server pod to run a clustered KIE Server environment. To scale a KIE Server, you must ensure that it uses a database server in a separate pod or an external database server, and not a built-in H2 database engine.

In a clustered deployment, several instances of the KIE Server run the same services. These servers can connect to the Business Central Controller using the same server ID, so they can receive the same requests from the controller. Red Hat OpenShift Container Platform provides load-balancing between the servers. Decision services and business optimizer services that run on a clustered KIE Server must be stateless, because requests from the same client might be processed by different instances.

You can also deploy several independent KIE Servers to run different services. In this case, the servers connect to the Business Central Controller with different server ID values. You can use the Business Central UI to deploy services to each of the servers.

Smart Router

The optional Smart Router component provides a layer between client applications and KIE Servers. It can be useful if you are using several independent KIE Servers.

The client application can use services running on different KIE Servers, but always connects to the Smart Router. The Smart Router automatically passes the request to the KIE Servers that runs the required service. The Smart Router also enables management of service versions and provides an additional load-balancing layer.

High-availability authoring environment

In a high-availability (HA) authoring environment, the Business Central pod is scaled, so several instances of Business Central are running. Red Hat OpenShift Container Platform provides load balancing for user requests. This environment provides optimal performance for multiple users and supports failover.

Each instance of Business Central includes the Maven repository for the built artifacts and uses the **.niogit** Git repository for source code. The instances use shared persistent storage for the repositories. A persistent volume with **ReadWriteMany** access is required for this storage.

An instance of Red Hat DataGrid provides indexing of all projects and assets developed in Business Central.

An instance of Red Hat AMQ propagates Java CDI messages between all instances of Business Central. For example, when a new project is created or when an asset is locked or modified on one of the instances, this information is immediately reflected in all other instances.

The controller strategy is not suitable for clustered deployment. In an OpenShift deployment, a high-availability Business Central must manage KIE Servers using the *OpenShift startup strategy*.

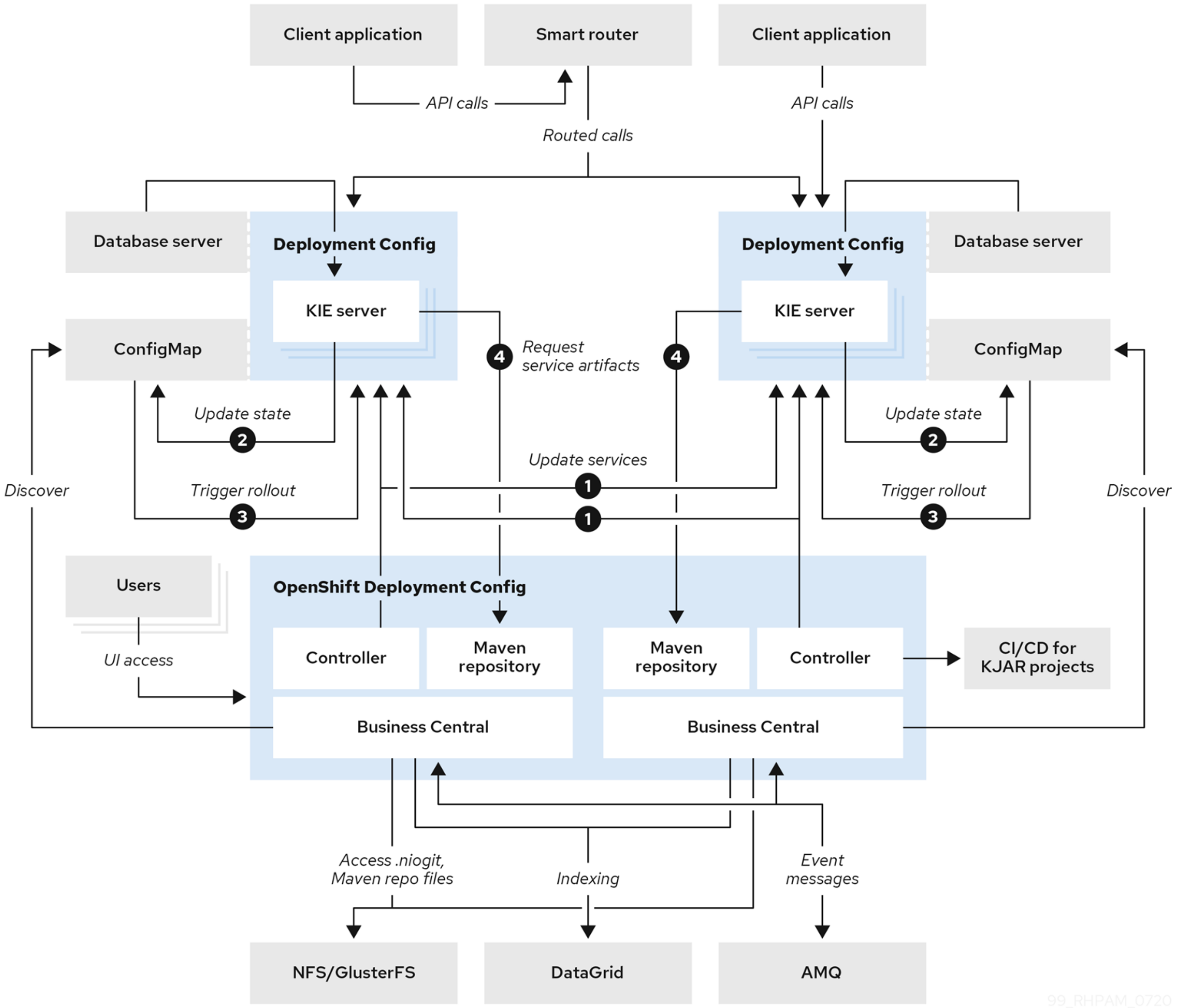
Each KIE Server deployment (which can be scaled) creates a ConfigMap that reflects its current state. The Business Central discovers all KIE Servers by reading their ConfigMaps.

When the user requests a change in KIE Server configuration (for example, deploys or undeploys a service), Business Central initiates a connection to the KIE Server and sends a REST API request. The KIE Server changes the ConfigMap to reflect the new configuration state and then triggers its own redeployment, so that all instances are redeployed and reflect the new configuration.

You can deploy several independent KIE Servers in your OpenShift environment. Each of the KIE Servers has a separate ConfigMap with the necessary configuration. You can scale each of the KIE Servers separately.

You can include Smart Router in the OpenShift deployment.

Figure 2.2. Architecture diagram for a high-availability authoring environment



CHAPTER 3. PREPARING TO DEPLOY RED HAT PROCESS AUTOMATION MANAGER IN YOUR OPENSHIFT ENVIRONMENT

Before deploying Red Hat Process Automation Manager in your OpenShift environment, you must complete several tasks. You do not need to repeat these tasks if you want to deploy additional images, for example, for new versions of processes or for other processes.

3.1. ENSURING YOUR ENVIRONMENT IS AUTHENTICATED TO THE RED HAT REGISTRY

To deploy Red Hat Process Automation Manager components of Red Hat OpenShift Container Platform, you must ensure that OpenShift can download the correct images from the Red Hat registry.

OpenShift must be configured to authenticate with the Red Hat registry using your service account user name and password. This configuration is specific for a namespace, and if operators work, the configuration is already completed for the **openshift** namespace.

However, if the image streams for Red Hat Process Automation Manager are not found in the **openshift** namespace or if the operator is configured to update Red Hat Process Automation Manager to a new version automatically, the operator needs to download images into the namespace of your project. You must complete the authentication configuration for this namespace.

Procedure

1. Ensure you are logged in to OpenShift with the **oc** command and that your project is active.

2. Complete the steps documented in Registry Service Accounts for Shared Environments . You must log in to Red Hat Customer Portal to access the document and to complete the steps to create a registry service account.

3. Select the **OpenShift Secret** tab and click the link under **Download secret** to download the YAML secret file.

4. View the downloaded file and note the name that is listed in the **name:** entry.

5. Run the following commands:

```
oc create -f <file_name>.yaml
oc secrets link default <secret_name> --for=pull
oc secrets link builder <secret_name> --for=pull
```

   Replace **<file_name>** with the name of the downloaded file and **<secret_name>** with the name that is listed in the **name:** entry of the file.

3.2. CREATING THE SECRETS FOR KIE SERVER

OpenShift uses objects called *secrets* to hold sensitive information such as passwords or keystores. For more information about OpenShift secrets, see What is a secret in the Red Hat OpenShift Container Platform documentation.

In order to provide HTTPS access, KIE Server uses an SSL certificate. The deployment can create a sample secret automatically. However, in production environments you must create an SSL certificate for KIE Server and provide it to your OpenShift environment as a secret.

Procedure

1. Generate an SSL keystore with a private and public key for SSL encryption for KIE Server. For more information on how to create a keystore with self-signed or purchased SSL certificates, see Generate a SSL Encryption Key and Certificate .

   NOTE

   In a production environment, generate a valid signed certificate that matches the expected URL for KIE Server.

2. Save the keystore in a file named **keystore.jks**.
3. Record the name of the certificate. The default value for this name in Red Hat Process Automation Manager configuration is **jboss**.
4. Record the password of the keystore file. The default value for this name in Red Hat Process Automation Manager configuration is **mykeystorepass**.
5. Use the **oc** command to generate a secret named **kieserver-app-secret** from the new keystore file:

```
$ oc create secret generic kieserver-app-secret --from-file=keystore.jks
```

3.3. CREATING THE SECRETS FOR BUSINESS CENTRAL

In order to provide HTTPS access, Business Central uses an SSL certificate. The deployment can create a sample secret automatically. However, in production environments you must create an SSL certificate for Business Central and provide it to your OpenShift environment as a secret.

Do not use the same certificate and keystore for Business Central and KIE Server.

Procedure

1. Generate an SSL keystore with a private and public key for SSL encryption for Business Central. For more information on how to create a keystore with self-signed or purchased SSL certificates, see Generate a SSL Encryption Key and Certificate .

   NOTE

   In a production environment, generate a valid signed certificate that matches the expected URL for Business Central.

2. Save the keystore in a file named **keystore.jks**.
3. Record the name of the certificate. The default value for this name in Red Hat Process Automation Manager configuration is **jboss**.
4. Record the password of the keystore file. The default value for this name in Red Hat Process Automation Manager configuration is **mykeystorepass**.

5. Use the **oc** command to generate a secret named **businesscentral-app-secret** from the new keystore file:

```
$ oc create secret generic businesscentral-app-secret --from-file=keystore.jks
```

3.4. CREATING THE SECRETS FOR THE AMQ BROKER CONNECTION

If you want to connect any KIE Server to an AMQ broker and to use SSL for the AMQ broker connection, you must create an SSL certificate for the connection and provide it to your OpenShift environment as a secret.

Procedure

1. Generate an SSL keystore with a private and public key for SSL encryption for the AMQ broker connection. For more information on how to create a keystore with self-signed or purchased SSL certificates, see Generate a SSL Encryption Key and Certificate .

   > NOTE
   >
   > In a production environment, generate a valid signed certificate that matches the expected URL for the AMQ broker connection.

2. Save the keystore in a file named **keystore.jks**.

3. Record the name of the certificate. The default value for this name in Red Hat Process Automation Manager configuration is **jboss**.

4. Record the password of the keystore file. The default value for this name in Red Hat Process Automation Manager configuration is **mykeystorepass**.

5. Use the **oc** command to generate a secret named **broker-app-secret** from the new keystore file:

```
$ oc create secret generic broker-app-secret --from-file=keystore.jks
```

3.5. CREATING THE SECRETS FOR SMART ROUTER

In order to provide HTTPS access, Smart Router uses an SSL certificate. The deployment can create a sample secret automatically. However, in production environments you must create an SSL certificate for Smart Router and provide it to your OpenShift environment as a secret.

Do not use the same certificate and keystore for Smart Router as the ones used for KIE Server or Business Central.

Procedure

1. Generate an SSL keystore with a private and public key for SSL encryption for Smart Router. For more information on how to create a keystore with self-signed or purchased SSL certificates, see Generate a SSL Encryption Key and Certificate .

NOTE

In a production environment, generate a valid signed certificate that matches the expected URL for Smart Router.

2. Save the keystore in a file named **keystore.jks**.

3. Record the name of the certificate. The default value for this name in Red Hat Process Automation Manager configuration is **jboss**.

4. Record the password of the keystore file. The default value for this name in Red Hat Process Automation Manager configuration is **mykeystorepass**.

5. Use the **oc** command to generate a secret named **smartrouter-app-secret** from the new keystore file:

```
$ oc create secret generic smartrouter-app-secret --from-file=keystore.jks
```

3.6. BUILDING A CUSTOM KIE SERVER EXTENSION IMAGE FOR AN EXTERNAL DATABASE

If you want to use an external database server for a KIE Server and the database server is not a MySQL or PostgreSQL server, you must build a custom KIE Server extension image with drivers for this server before deploying your environment.

Complete the steps in this build procedure to provide drivers for any of the following database servers:

- Microsoft SQL Server
- IBM DB2
- Oracle Database
- Sybase

Optionally, you can use this procedure to build a new version of drivers for any of the following database servers:

- MySQL
- MariaDB
- PostgreSQL

For the supported versions of the database servers, see Red Hat Process Automation Manager 7 Supported Configurations.

The build procedure creates a custom extension image that extends the existing KIE Server image. You must import this custom extension image into your OpenShift environment and then reference it in the **EXTENSIONS_IMAGE** parameter.

Prerequisites

- You are logged in to your OpenShift environment using the **oc** command. Your OpenShift user must have the **registry-editor** role.

- For Oracle Database, IBM DB2, or Sybase, you downloaded the JDBC driver from the database server vendor.
- You have installed the following required software:
  - Docker: For installation instructions, see Get Docker.
  - Cekit version 3.2: For installation instructions, see Installation.
  - The following libraries and extensions for Cekit. For more information, see Dependencies.
    - **docker**, provided by the **python3-docker** package or similar package
    - **docker-squash**, provided by the **python3-docker-squash** package or similar package
    - **behave**, provided by the **python3-behave** package or similar package

Procedure

1. For IBM DB2, Oracle Database, or Sybase, provide the JDBC driver JAR file in a local directory.
2. Download the **rhpam-7.8.0-openshift-templates.zip** product deliverable file from the Software Downloads page of the Red Hat Customer Portal.
3. Unzip the file and, using the command line, change to the **templates/contrib/jdbc/cekit** directory of the unzipped file. This directory contains the source code for the custom build.
4. Enter one of the following commands, depending on the database server type:
   - For Microsoft SQL Server:

     ```
     make mssql
     ```

   - For MySQL:

     ```
     make mysql
     ```

   - For PostgreSQL:

     ```
     make postgresql
     ```

   - For MariaDB:

     ```
     make mariadb
     ```

   - For IBM DB2:

     ```
     make db2 artifact=/tmp/db2jcc4.jar version=10.2
     ```

     In this command, replace **/tmp/db2jcc4.jar** with the path name of the IBM DB2 driver and **10.2** with the version of the driver.

   - For Oracle Database:

     ```
     make oracle artifact=/tmp/ojdbc7.jar version=7.0
     ```

   In this command, replace **/tmp/ojdbc7.jar** with the path name of the Oracle Database driver and **7.0** with the version of the driver.

   - For Sybase:

     ```
     make build sybase artifact=/tmp/jconn4-16.0_PL05.jar version=16.0_PL05
     ```

     In this command, replace **/tmp/jconn4-16.0_PL05.jar** with the path name of the downloaded Sybase driver and **16.0_PL05** with the version of the driver.

     Alternatively, if you need to update the driver class or driver XA class for the Sybase driver, you can set the **DRIVER_CLASS** or **DRIVER_XA_CLASS** variable for this command, for example:

     ```
     export DRIVER_CLASS=another.class.Sybase && make sybase artifact=/tmp/jconn4-
     16.0_PL05.jar version=16.0_PL05
     ```

5. Enter the following command to list the Docker images that are available locally:

   ```
   docker images
   ```

   Note the name of the image that was built, for example, **jboss-kie-db2-extension-openshift-image**, and the version tag of the image, for example, **11.1.4.4** (not the **latest** tag).

6. Access the registry of your OpenShift environment directly and push the image to the registry. Depending on your user permissions, you can push the image into the **openshift** namespace or into a project namespace. For instructions about accessing the registry and pushing the images, see Accessing registry directly from the cluster in the Red Hat OpenShift Container Platform product documentation.

3.7. PREPARING GIT HOOKS

In an authoring environment you can use Git hooks to execute custom operations when the source code of a project in Business Central is changed. The typical use of Git hooks is for interaction with an upstream repository.

To enable Git hooks to interact with an upstream repository using SSH authentication, you must also provide a secret key and a known hosts file for authentication with the repository.

Skip this procedure if you do not want to configure Git hooks.

Procedure

1. Create the Git hooks files. For instructions, see the Git hooks reference documentation.

   NOTE

   A **pre-commit** script is not supported in Business Central. Use a **post-commit** script.

2. Create a configuration map (ConfigMap) or persistent volume with the files.

   - If the Git hooks consist of one or several fixed script files, use the **oc** command to create a configuration map. For example:

```
oc create configmap git-hooks --from-file=post-commit=post-commit
```

- If the Git hooks consist of long files or depend on binaries, such as executable or JAR files, use a persistent volume. You must create a persistent volume, create a persistent volume claim and associate the volume with the claim, and transfer files to the volume.
  For instructions about persistent volumes and persistent volume claims, see Storage in the Red Hat OpenShift Container Platform documentation. For instructions about copying files onto a persistent volume, see Transferring files in and out of containers .

3. If the Git hooks scripts must interact with an upstream repository using SSH authentication, prepare a secret with the necessary files:

   a. Prepare the **id_rsa** file with a private key that matches a public key stored in the repository.

   b. Prepare the **known_hosts** file with the correct name, address, and public key for the repository.

   c. Create a secret with the two files using the **oc** command, for example:

   ```
   oc create secret git-hooks-secret --from-file=id_rsa=id_rsa --from-file=known_hosts=known_hosts
   ```

   NOTE

   When the deployment uses this secret, it mounts the **id_rsa** and **known_hosts** files into the **/home/jboss/.ssh** directory on Business Central pods.

3.8. PROVISIONING PERSISTENT VOLUMES WITH READWRITEMANY ACCESS MODE USING NFS

If you want to deploy Business Central Monitoring or high-availability Business Central, your environment must provision persistent volumes with **ReadWriteMany** access mode.

If your configuration requires provisioning persistent volumes with **ReadWriteMany** access mode but your environment does not support such provisioning, use NFS to provision the volumes. Otherwise, skip this procedure.

Procedure

Deploy an NFS server and provision the persistent volumes using NFS. For information about provisioning persistent volumes using NFS, see the "Persistent storage using NFS" section of the OpenShift Container Platform 4.3 Storage guide.

3.9. EXTRACTING THE SOURCE CODE FROM BUSINESS CENTRAL FOR USE IN AN S2I BUILD

If you are planning to create immutable KIE servers using the source-to-image (S2I) process, you must provide the source code for your services in a Git repository. If you are using Business Central for authoring services, you can extract the source code for your service and place it into a separate Git repository, such as GitHub or an on-premise installation of GitLab, for use in the S2I build.

Skip this procedure if you are not planning to use the S2I process or if you are not using Business Central for authoring services.

Procedure

1. Use the following command to extract the source code:

   ```
   git clone https://<business-central-host>:443/git/<MySpace>/<MyProject>
   ```

   In this command, replace the following variables:

   - **<business-central-host>** with the host on which Business Central is running
   - **<MySpace>** with the name of the Business Central space in which the project is located
   - **<MyProject>** with the name of the project

   NOTE

   To view the full Git URL for a project in Business Central, click **Menu** → **Design** → ***<MyProject>*** → **Settings**.

   NOTE

   If you are using self-signed certificates for HTTPS communication, the command might fail with an **SSL certificate problem** error message. In this case, disable SSL certificate verification in **git**, for example, using the **GIT_SSL_NO_VERIFY** environment variable:

   ```
   env GIT_SSL_NO_VERIFY=true git clone https://<business-central-host>:443/git/<MySpace>/<MyProject>
   ```

2. Upload the source code to another Git repository, such as GitHub or GitLab, for the S2I build.

3.10. PREPARING FOR DEPLOYMENT IN A RESTRICTED NETWORK

You can deploy Red Hat Process Automation Manager in a restricted network that is not connected to the public Internet. For instructions about operator deployment in a restricted network, see Using Operator Lifecycle Manager on restricted networks in Red Hat OpenShift Container Platform documentation.

IMPORTANT

In Red Hat Process Automation Manager 7.8, deployment on restricted networks is for Technology Preview only. For more information on Red Hat Technology Preview features, see Technology Preview Features Scope .

In order to use a deployment that does not have outgoing access to the public Internet, you must also prepare a Maven repository with a mirror of all the necessary artifacts. For instructions about creating this repository, see Section 3.11, "Preparing a Maven mirror repository for offline use" .

3.11. PREPARING A MAVEN MIRROR REPOSITORY FOR OFFLINE USE

If your Red Hat OpenShift Container Platform environment does not have outgoing access to the public Internet, you must prepare a Maven repository with a mirror of all the necessary artifacts and make this repository available to your environment.

NOTE

You do not need to complete this procedure if your Red Hat OpenShift Container Platform environment is connected to the Internet.

Prerequisites

- A computer that has outgoing access to the public Internet is available.

Procedure

1. Configure a Maven release repository to which you have write access. The repository must allow read access without authentication and your OpenShift environment must have network access to this repository.
   You can deploy a Nexus repository manager in the OpenShift environment. For instructions about setting up Nexus on OpenShift, see Setting up Nexus in the Red Hat OpenShift Container Platform 3.11 documentation. The documented procedure is applicable to Red Hat OpenShift Container Platform version 4. Use this repository as a separate mirror repository.

   Alternatively, if you use a custom external repository (for example, Nexus) for your services, you can use the same repository as a mirror repository.

2. On the computer that has an outgoing connection to the public Internet, complete the following steps:

   a. Click **Red Hat Process Automation Manager 7.8.0 Offliner Content List**to download the **rhpam-7.8.0-offliner.zip** product deliverable file from the Software Downloads page of the Red Hat Customer Portal.

   b. Extract the contents of the **rhpam-7.8.0-offliner.zip** file into any directory.

   c. Change to the directory and enter the following command:

   ```
   ./offline-repo-builder.sh offliner.txt
   ```

   This command creates a **repository** subdirectory and downloads the necessary artifacts into this subdirectory.

   If a message reports that some downloads have failed, run the same command again. If downloads fail again, contact Red Hat support.

   d. Upload all artifacts from the **repository** subdirectory to the Maven mirror repository that you prepared. You can use the Maven Repository Provisioner utility, available from the Maven repository tools Git repository, to upload the artifacts.

3. If you developed services outside Business Central and they have additional dependencies, add the dependencies to the mirror repository. If you developed the services as Maven projects, you can use the following steps to prepare these dependencies automatically. Complete the steps on the computer that has an outgoing connection to the public Internet.

   a. Create a backup of the local Maven cache directory (**~/.m2/repository**) and then clear the directory.

   b. Build the source of your projects using the **mvn clean install** command.

c. For every project, enter the following command to ensure that Maven downloads all runtime dependencies for all the artifacts generated by the project:

```
mvn -e -DskipTests dependency:go-offline -f /path/to/project/pom.xml --batch-mode -
Djava.net.preferIPv4Stack=true
```

Replace **/path/to/project/pom.xml** with the correct path to the **pom.xml** file of the project.

d. Upload all artifacts from the local Maven cache directory (~/**.m2/repository**) to the Maven mirror repository that you prepared. You can use the Maven Repository Provisioner utility, available from the Maven repository tools Git repository, to upload the artifacts.

CHAPTER 4. DEPLOYMENT AND MANAGEMENT OF A RED HAT PROCESS AUTOMATION MANAGER ENVIRONMENT USING OPENSHIFT OPERATORS

To deploy a Red Hat Process Automation Manager environment, the OpenShift operator uses a YAML source that describes the environment. Red Hat Process Automation Manager provides an installer that you can use to form the YAML source and deploy the environment.

When the Business Automation operator deploys the environment, it creates a YAML description of the environment, and then ensures that the environment is consistent with the description at all times. You can edit the description to modify the environment.

You can remove the environment by deleting the operator application in Red Hat OpenShift Container Platform.

NOTE

When you remove an environment with a high-availability Business Central, the operator does not delete Persistent Volume Claims that were created as part of the JBoss Datagrid and JBoss AMQ StatefulSet creation. This behaviour is a part of Kubernetes design, as deletion of the Persistent Volume Claims could cause data loss. For more information about handling persistent volumes during deletion of a StatefulSet, see the Kubernetes documentation.

If you create a new environment using the same namespace and the same application name, the environment reuses the persistent volumes for increased performance.

You can delete the Persistent Volume Claims manually.

4.1. SUBSCRIBING TO THE BUSINESS AUTOMATION OPERATOR

To be able to deploy Red Hat Process Automation Manager using operators, you must subscribe to the Business Automation operator in OpenShift.

Procedure

1. Enter your project in the OpenShift Web cluster console.
2. In the OpenShift Web console navigation panel, select **Catalog → OperatorHub** or **Operators → OperatorHub**.
3. Search for **Business Automation**, select it and click **Install**.
4. On the **Create Operator Subscription** page, select your target namespace and approval strategy.
   Optional: Set **Approval strategy** to **Automatic** to enable automatic operator updates. An operator update does not immediately update the product, but is required before you update the product. Configure automatic or manual product updates using the settings in every particular product deployment.
5. Click **Subscribe** to create a subscription.

4.2. DEPLOYING A RED HAT PROCESS AUTOMATION MANAGER ENVIRONMENT USING THE OPERATOR

After you subscribe to the Business Automation operator, you can use the installer wizard to configure and deploy a Red Hat Process Automation Manager environment.

IMPORTANT

In Red Hat Process Automation Manager 7.8, the operator installer wizard is for Technology Preview only. For more information on Red Hat Technology Preview features, see Technology Preview Features Support Scope .

4.2.1. Starting the deployment of a Red Hat Process Automation Manager environment using the Business Automation operator

To start deploying a Red Hat Process Automation Manager environment using the Business Automation operator, access the installer wizard. The installer wizard is deployed when you subscribe to the operator.

Prerequisites

- You subscribed to the Business Automation operator. For instructions about subscribing to the operator, see Section 4.1, "Subscribing to the Business Automation operator" .

Procedure

1. In the Red Hat OpenShift Container Platform web cluster console menu, select **Catalog → Installed operators** or **Operators → Installed operators**.

2. Click the name of the operator that contains **businessautomation**. Information about this operator is displayed.

3. Click the **Installer** link located on the right side of the window.

4. If prompted, log in with your OpenShift credentials.

Result

The **Installation** tab of the wizard is displayed.

4.2.2. Setting the basic configuration of the environment

After you start to deploy a Red Hat Process Automation Manager environment using the Business Automation operator, you must select the type of the environment and set other basic configuration.

Prerequisites

- You started to deploy a Red Hat Process Automation Manager environment using the Business Automation operator and accessed the installer wizard according to the instructions in Section 4.2.1, "Starting the deployment of a Red Hat Process Automation Manager environment using the Business Automation operator".

Procedure

1. In the **Application Name** field, enter a name for the OpenShift application. This name is used in the default URLs for all components.

2. In the **Environment** list, select the type of environment. This type determines the default configuration; you can modify this configuration as necessary. The following types are available for Red Hat Process Automation Manager:

   - **rhpam-trial**: A trial environment that you can set up quickly and use to evaluate or demonstrate developing and running assets. Includes Business Central and a KIE Server. This environment does not use any persistent storage, and any work you do in the environment is not saved.

   - **rhpam-authoring**: An environment for creating and modifying services using Business Central. It consists of pods that provide Business Central for the authoring work and a KIE Server for test execution of the services.

   - **rhpam-authoring-ha**: An environment for creating and modifying services using Business Central. It consists of pods that provide Business Central for the authoring work and a KIE Server for test execution of the services. This version of the authoring environment supports scaling the Business Central pod to ensure high availability.

     IMPORTANT

     In Red Hat Process Automation Manager 7.8, high-availability Business Central functionality deployment using the operator is for Technology Preview only. For more information about Red Hat Technology Preview features, see Technology Preview Features Support Scope . For a fully supported high-availabilityAvailability deployment, use the high-availability authoring template on Red Hat OpenShift Container Platform version 3.11. For instructions about deploying this template, see *Deploying a Red Hat Process Automation Manager authoring environment on Red Hat OpenShift Container Platform*.

   - **rhpam-production**: An environment for running existing services for staging and production purposes. This environment includes Business Central Monitoring, Smart Router, and two groups of KIE Server pods. You can deploy and undeploy services on every such group and also scale the group up or down as necessary. Use Business Central Monitoring to deploy, run, and stop the services and to monitor their execution.

   - **rhpam-production-immutable**: An alternate environment for running existing services for staging and production purposes. You can configure one or more KIE Server pods that build services from source or pull them from a Maven repository. You can then replicate each pod as necessary.
     You cannot remove any service from the pod or add any new service to the pod. If you want to use another version of a service or to modify the configuration in any other way, deploy a new server image to replace the old one. You can use any container-based integration workflows to manage the pods.

     When configuring this environment, in the **KIE Servers** tab you must customize the KIE Server and either click the **Set immutable server configuration** button or set the **KIE_SERVER_CONTAINER_DEPLOYMENT** environment variable. For instructions about configuring the KIE Server, see Section 4.2.5, “Setting custom KIE Server configuration of the environment”.

     Optionally, you can also use the **Console** tab to include Business Central Monitoring in this environment to monitor, stop, and restart the execution of process services. For instructions about configuring Business Central Monitoring, see Section 4.2.4, “Setting the Business Central configuration of the environment”.

3. If you want to enable automatic upgrades to new versions, select the **Enable Upgrades** box. If this box is selected, when a new patch version of Red Hat Process Automation Manager 7.8 becomes available, the operator automatically upgrades your deployment to this version. All services are preserved and normally remain available throughout the upgrade process.
   If you also want to enable the same automatic upgrade process when a new minor version of Red Hat Process Automation Manager 7.x becomes available, select the **Include minor version upgrades** box.

   > NOTE
   >
   > Disable automatic updates if you want to use a custom image for any component of Red Hat Process Automation Manager.

4. Optional: If you want to use image tags for downloading images, select the **Use Image Tags** box. This setting is useful if you use a custom registry or if you are directed by Red Hat support.

5. If you want to use a custom image registry, under **Custom registry**, enter the URL of the registry in the **Image registry** field. If this registry does not have a properly signed and recognized SSL certificate, select the **Insecure** box.

6. Under **Admin user**, enter the user name and password for the administrative user for Red Hat Process Automation Manager in the **Username** and **Password** fields.

   > IMPORTANT
   >
   > If you use RH-SSO or LDAP authentication, the same user must be configured in your authentication system with the **kie-server,rest-all,admin** roles for Red Hat Process Automation Manager.

7. If you want to use a custom version tag for images, complete the following steps:

   a. Click **Next** to access the **Security** tab.

   b. Scroll to the bottom of the window.

   c. Enter the image tag in the **Image tag** field.

Next steps

If you want to deploy the environment with the default configuration, click **Finish** and then click **Deploy** to deploy the environment. Otherwise, continue to set other configuration parameters.

4.2.3. Setting the security configuration of the environment

After you set the basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator, you can optionally configure authentication (security) settings for the environment.

Prerequisites

- You completed basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator in the installer wizard according to the instructions in Section 4.2.2, "Setting the basic configuration of the environment".

- If you want to use RH-SSO or LDAP for authentication, you created users with the correct roles in your authentication system. You must create at least one administrative user (for example, **adminUser**) with the **kie-server,rest-all,admin** roles. This user must have the user name and password that you configured on the **Installation** tab.

- If you want to use RH-SSO authentication, you created the clients in your RH-SSO system for all components of your environment, specifying the correct URLs. This action ensures maximum control. Alternatively, the deployment can create the clients.

Procedure

1. If the **Installation** tab is open, click **Next** to view the **Security** tab.

2. In the **Authentication mode** list, select one of the following modes:

   - **Internal**: You configure the initial administration user when deploying the environment. The user can use Business Central to set up other users as necessary.

   - **RH-SSO**: Red Hat Process Automation Manager uses Red Hat Single Sign-On for authentication.

   - **LDAP**: Red Hat Process Automation Manager uses LDAP for authentication

3. Complete the security configuration based on the **Authentication mode** that you selected.
   If you selected **RH-SSO**, configure RH-SSO authentication:

   a. In the **RH-SSO URL** field, enter the RH-SSO URL.

   b. In the **Realm** field, enter the RH-SSO realm name.

   c. If you did not create RH-SSO clients for components of your environment enter the credentials of an administrative user for your RH-SSO system in the **SSO admin user** and **SSO admin password** fields.

   d. If your RH-SSO system does not have a proper signed SSL certificate, select the **Disable SSL cert validation** box.

   e. If you want to change the RH-SSO principal attribute used for the user name, in the **Principal attribute** field enter the name of the new attribute.

   If you selected **LDAP**, configure LDAP authentication:

   a. In the **LDAP URL** field, enter the LDAP URL.

   b. Configure LDAP parameters that correspond to the settings of the LdapExtended Login module of Red Hat JBoss EAP. For instructions about using these settings, see LdapExtended Login Module.

4. If you selected **RH-SSO** or **LDAP**, if your RH-SSO or LDAP system does not define all the roles required for your deployment, you can map authentication system roles to Red Hat Process Automation Manager roles.

To enable role mapping, you must provide a role mapping configuration file in an OpenShift configuration map or secret object in the project namespace. The file must contain entries in the following format:

```
ldap_role = product_role1, product_role2...
```

For example:

```
admins = kie-server,rest-all,admin
```

To enable the use of this file, make the following changes:

a. Under **RoleMapper**, in the **Roles properties file** field, enter the fully qualified path name of the role mapping configuration file, for example, **/opt/eap/standalone/configuration/rolemapping/rolemapping.properties**.

b. If you want to replace roles defined in the authentication system with roles that you define in the mapping file, select the **Replace roles** box. Otherwise, both the roles defined in RH-SSO or LDAP and the roles defined in the configuration file are available.

c. In the fields under **RoleMapper Configuration object**, select the **Kind** of the object that provides the file (**ConfigMap** or **Secret**) and enter the **Name** of the object. This object is automatically mounted on Business Central and KIE Server pods in the path that you specified for the role mapping configuration file.

5. Configure other passwords, if necessary:

   - **AMQ password** and **AMQ cluster password** are passwords for interaction with ActiveMQ using the JMS API.

   - **Keystore password** is the password for the keystore files used in secrets for HTTPS communication. Set this password if you created secrets according to instructions in Section 3.2, “Creating the secrets for KIE Server” or Section 3.3, “Creating the secrets for Business Central”.

   - **Database password** is the password for database server pods that are a part of the environments.

Next steps

If you want to deploy the environment with the default configuration of all components, click **Finish** and then click **Deploy** to deploy the environment. Otherwise, continue to set configuration parameters for Business Central, KIE Servers, and Smart Router.

4.2.4. Setting the Business Central configuration of the environment

After you set the basic and security configuration of a Red Hat Process Automation Manager environment using the Business Automation operator, you can optionally configure settings for the Business Central or Business Central Monitoring component of the environment.

Prerequisites

- You completed basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator in the installer wizard according to the instructions in Section 4.2.2, “Setting the basic configuration of the environment”.

- If you want to use RH-SSO or LDAP for authentication, you completed security configuration according to the instructions in Section 4.2.3, “Setting the security configuration of the environment”.

Procedure

1. If the **Installation** or **Security** tab is open, click **Next** until you view the **Console** tab.

2. If you created the secret for Business Central according to the instructions in Section 3.3, “Creating the secrets for Business Central”, enter the name of the secret in the **Secret** field.

3. Optionally, configure Git hooks.
   In an authoring environment, you can use Git hooks to facilitate interaction between the internal Git repository of Business Central and an external Git repository. If you want to use Git hooks, you must prepare a Git hooks directory in an OpenShift configuration map, secret, or persistent volume claim object in the project namespace. You can also prepare a secret with the SSH key and known hosts files for Git SSH authentication. For instructions about preparing Git hooks, see Section 3.7, “Preparing Git hooks”.

   To use a Git hooks directory, make the following changes:

   a. Under **GitHooks**, in the **Mount path** field, enter a fully qualified path for the directory, for example, **/opt/kie/data/git/hooks**.

   b. In the fields under **GitHooks Configuration object**, select the **Kind** of the object that provides the file (**ConfigMap**, **Secret**, or **PersistentVolumeClaim**) and enter the **Name** of the object. This object is automatically mounted on the Business Central pods in the path that you specified for the Git hooks directory.

   c. Optionally, in the **SSH secret** field enter the name of the secret with the SSH key and known hosts files.

4. Optionally, enter the number of replicas for Business Central or Business Central monitoring in the **Replicas** field. Do not change this number in a **rhpam-authoring** environment.

5. Optionally, enter requested and maximum CPU and memory limits in the fields under **Resource quotas**.

6. If you want to customize the configuration of the Java virtual machine on the Business Central pods, select the **Enable JVM configuration** box and then enter information in any of the fields under **Enable JVM configuration**. All fields are optional. For the JVM parameters that you can configure, see Section 4.4, “JVM configuration parameters” .

7. If you selected RH-SSO authentication, configure RH-SSO for Business Central:

   a. Enter the client name in the **Client name** field and the client secret in the **Client secret** field. If a client with this name does not exist, the deployment attempts to create a new client with this name and secret.

   b. If the deployment is to create a new client, enter the HTTP and HTTPS URLs that will be used for accessing the KIE Server into the **SSO HTTP URL** and **SSO HTTPS URL** fields. This information is recorded in the client.

8. Optionally, depending on your needs, set environment variables. To set an environment variable, click **Add new Environment variable**, then enter the name and value for the variable in the **Name** and **Value** fields.

- In a **rhpam-production** or **rhpam-production-immutable** environment, if you want Business Central Monitoring to run in a simplified mode that does not use a file system, set the **ORG_APPFORMER_SIMPLIFIED_MONITORING_ENABLED** to **true**.
  In the simplified mode, Business Central Monitoring does not require a persistent volume claim. You can use this mode in environments that do not support **ReadWriteMany** access to persistent storage. You can not use Business Central Monitoring in the simplified mode to design custom dashboards.

- If you want to use an external Maven repository, set the following variables:

  - **MAVEN_REPO_URL**: The URL for the Maven repository

  - **MAVEN_REPO_ID**: An identifier for the Maven repository, for example, **repo-custom**

  - **MAVEN_REPO_USERNAME**: The user name for the Maven repository

  - **MAVEN_REPO_PASSWORD** The password for the Maven repository

  IMPORTANT

  In an authoring environment, if you want Business Central to push a project into an external Maven repository, you must configure this repository during deployment and also configure exporting to the repository in every project. For information about exporting Business Central projects to an external Maven repository, see *Packaging and deploying a Red Hat Process Automation Manager project*.

- If your OpenShift environment does not have a connection to the public Internet, configure access to a Maven mirror that you set up according to Section 3.11, "Preparing a Maven mirror repository for offline use". Set the following variables:

  - **MAVEN_MIRROR_URL**: The URL for the Maven mirror repository that you set up in Section 3.11, "Preparing a Maven mirror repository for offline use" . This URL must be accessible from a pod in your OpenShift environment.

  - **MAVEN_MIRROR_OF**: The value that determines which artifacts are to be retrieved from the mirror. For instructions about setting the **mirrorOf** value, see Mirror Settings in the Apache Maven documentation. The default value is **external:***. With this value, Maven retrieves every required artifact from the mirror and does not query any other repositories.
    If you configure an external Maven repository (**MAVEN_REPO_URL**), change **MAVEN_MIRROR_OF** to exclude the artifacts in this repository from the mirror, for example, **external:*,!repo-custom**. Replace **repo-custom** with the ID that you configured in **MAVEN_REPO_ID**.

    If your authoring environment uses a built-in Business Central Maven repository, change **MAVEN_MIRROR_OF** to exclude the artifacts in this repository from the mirror: **external:*,!repo-rhpamcentr**.

- In some authoring environments, you might need to ensure that several users can deploy services on the same KIE Server at the same time. By default, after deploying a service onto a KIE Server using Business Central, the user needs to wait for some seconds before more services can be deployed. The **OpenShiftStartupStrategy** setting is enabled by default and causes this limitation. To remove the limitation, you can configure an **rhpam-authoring** environment to use the *controller strategy*. Do not make this change unless a specific need for it exists; if you decide to enable controller strategy, make this change on Business Central and on all KIE Servers in the same environment.

> **NOTE**
>
> Do not enable the controller strategy in an environment with a high-availability Business Central. In such environments the controller strategy does not function correctly.

To enable the controller strategy on Business Central, set the **KIE_SERVER_CONTROLLER_OPENSHIFT_ENABLED** environment variable to **false**.

Next steps

If you want to deploy the environment with the default configuration of KIE Servers, without Smart Router, and without Process Instance Migration, click **Finish** and then click **Deploy** to deploy the environment. Otherwise, continue to set configuration parameters for KIE Servers and Smart Router.

4.2.5. Setting custom KIE Server configuration of the environment

Every environment type in the Business Automation operator includes one or several KIE Servers by default.

Optionally, you can set custom configuration for KIE Servers. In this case, default KIE Servers are not created and only the KIE Servers that you configure are deployed.

Prerequisites

- You completed basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator in the installer wizard according to the instructions in Section 4.2.2, “Setting the basic configuration of the environment”.
- If you want to use RH-SSO or LDAP for authentication, you completed security configuration according to the instructions in Section 4.2.3, “Setting the security configuration of the environment”.

Procedure

1. If the **Installation**, **Security**, or **Console** tab is open, click **Next** until you view the **KIE Servers** tab.
2. Click **Add new KIE Server** to add a new KIE Server configuration.
3. In the **Id** field, enter an identifier for the KIE Server. If the KIE Server connects to a Business Central or Business Central Monitoring instance, this identifier determines which server group the server joins.
4. In the **Name** field, enter a name for the KIE Server.
5. In the **Deployments** field, enter the number of similar KIE Servers that are to be deployed. The installer can deploy several KIE Servers with the same configuration. The identifiers and names of the KIE Servers are modified automatically and remain unique.

6. If you created the secret for KIE Server according to the instructions in Section 3.2, “Creating the secrets for KIE Server”, enter the name of the secret in the **Keystore secret** field.

7. Optionally, enter the number of replicas for the KIE Server in the **Replicas** field.

8. If you want to use a custom image for the KIE Server, complete the following additional steps:

   a. Click **Set KIE Server image**

   b. Enter the name of the image stream in the **Name** field.

   c. If the image stream is not in the **openshift** namespace, enter the namespace in the **Namespace** field.

   > NOTE
   >
   > Do not change the **Kind** value to **DockerImage**. This option does not work in Red Hat Process Automation Manager 7.8.0.

   For instructions about creating custom images, see Section 4.5, “Creating custom images for KIE Server”.

9. If you want to configure an immutable KIE Server using a Source to Image (S2I) build, complete the following additional steps:

   > IMPORTANT
   >
   > If you want to configure an immutable KIE Server that pulls services from the Maven repository, do not click **Set Immutable server configuration** and do not complete these steps. Instead, set the **KIE_SERVER_CONTAINER_REPLOYMENT** environment variable.

   a. Click **Set Immutable server configuration**.

   b. In the **KIE Server container deployment** field, enter the identifying information of the services (KJAR files) that the deployment must extract from the result of a Source to Image (S2I) build. The format is **<containerId>=<groupId>:<artifactId>:<version>** or, if you want to specify an alias name for the container, **<containerId>(<aliasId>)=<groupId>:<artifactId>:<version>**. You can provide two or more KJAR files using the **|** separator, as illustrated in the following example: **containerId=groupId:artifactId:version|c2(alias2)=g2:a2:v2**.

   c. If your OpenShift environment does not have a connection to the public Internet, enter the URL of the Maven mirror that you set up according to Section 3.11, “Preparing a Maven mirror repository for offline use” in the **Maven mirror URL** field.

   d. In the **Artifact directory** field, enter the path within the project that contains the required binary files (KJAR files and any other necessary files) after a successful Maven build. Normally this directory is the target directory of the build. However, you can provide prebuilt binaries in this directory in the Git repository.

   e. If you want to use a custom base KIE Server image for the S2I build, click **Set Base build image** and then enter the name of the image stream in the **Name** field. If the image stream is not in the **openshift** namespace, enter the namespace in the **Namespace** field. If you want to use a Docker image name and not an OpenShift image stream tag, change the **Kind** value to **DockerImage**.

f. Click **Set Git source** and enter information in the following fields:

- **S2I Git URI**: The URI for the Git repository that contains the source for your services.
- **Reference**: The branch in the Git repository.
- **Context directory**: (Optional) The path to the source within the project downloaded from the Git repository. By default, the root directory of the downloaded project is the source directory.

g. If you want to set a Git Webhook so changes in the Git repository cause an automatic rebuild of the KIE Server, click **Add new Webhook**. Select the type of the Webhook from the **Type** list and enter the secret string for the Webhook in the **Secret** field.

10. Optionally, enter requested and maximum CPU and memory limits in the fields under **Resource quotas**. If you are configuring several KIE Servers, the limits apply to each server separately.

11. If you selected RH-SSO authentication, configure RH-SSO for the KIE Server:

a. Enter the client name in the **Client name** field and the client secret in the **Client secret** field. If a client with this name does not exist, the deployment attempts to create a new client with this name and secret.

b. If the deployment is to create a new client, enter the HTTP and HTTPS URLs that will be used for accessing the KIE Server into the **SSO HTTP URL** and **SSO HTTPS URL** fields. This information is recorded in the client.

12. If you want to interact with the KIE Server through JMS API using an external AMQ message broker, enable the **Enable JMS Integration** setting. Additional fields for configuring JMS Integration are displayed and you must enter the values as necessary:

- **User name**, **Password**: The user name and password of a standard broker user, if user authentication in the broker is required in your environment.
- **Executor**: Select this setting to disable the JMS executor. The executor is enabled by default.
- **Executor transacted**: Select this setting to enable JMS transactions on the executor queue.
- **Enable signal**: Select this setting to enable signal configuration through JMS.
- **Enable audit**: Select this setting to enable audit logging through JMS.
- **Audit transacted**: Select this setting to enable JMS transactions on the audit queue.
- **Queue executor**, **Queue request**, **Queue response**, **Queue signal**, **Queue audit**: Custom JNDI names of the queues to use. If you set any of these values, you must also set the **AMQ queues** parameter.
- **AMQ Queues**: AMQ queue names, separated by commas. These queues are automatically created when the broker starts and are accessible as JNDI resources in the JBoss EAP server. If you are using any custom queue names, you must enter the names of all the queues uses by the server in this field.
- **Enable SSL integration**: Select this setting if you want to use an SSL connection to the AMQ broker. In this case you must also provide the name of the secret that you created in Section 3.4, “Creating the secrets for the AMQ broker connection” and the names and

passwords of the key store and trust store that you used for the secret.

13. If you want to customize the configuration of the Java virtual machine on the KIE Server pods, select the **Enable JVM configuration** box and then enter information in any of the fields under **Enable JVM configuration**. All fields are optional. For the JVM parameters that you can configure, see Section 4.4, "JVM configuration parameters" .

14. In the **Database type** field, select the database that the KIE Server must use. The following values are available:

    - **mysql**: A MySQL server, created in a separate pod.
    - **postgresql**: A PostgreSQL server, created in a separate pod. Use this setting unless you have a specific reason to use any other setting.
    - **h2**: A built-in **h2** database engine that does not require a separate pod. Do not scale the KIE Server pod if you use this setting.
    - **external**: An external database server.

15. If you selected any database except **external**, a Persistent Volume Claim will be created to store the database. Optionally, set configuration parameters for the persistent volume:

    - In the **Size** field, enter the size of the persistence volume.
    - In the **StorageClass name** field, enter the storage class name for the persistent volume.

16. Optionally, if you selected the **external** database, configure the KIE Server extension image. If you want to use any database server except PostgreSQL, MySQL, or MariaDB, you must provide a KIE Server extension image with the database server driver according to instructions in Section 3.6, "Building a custom KIE Server extension image for an external database" . To configure the KIE Server to use this extension image, make the following changes:

    a. Select the **Enable extension image stream** box.

    b. In the **Extension image stream tag** field, enter the ImageStreamTag definition for the image that you created, for example, **jboss-kie-db2-extension-openshift-image:11.1.4.4**

    c. Optionally, in the **Extension image stream namespace** field, enter the namespace into which you pushed the image. If you do not enter any value in this field, the operator expects the image to be in the **openshift** namespace.

    d. Optionally, in the **Extension image install directory** field, enter the directory within the extensions image where the extensions are located. If you used the procedure in Section 3.6, "Building a custom KIE Server extension image for an external database" to build the image, do not enter any value for this field.

17. If you selected an external database server, provide the following information in additional fields:

    a. **Driver**: Enter the database server driver, depending on the server type:

       - **mysql**
       - **postgresql**
       - **mariadb**
       - **mssql**

- **db2**
- **oracle**
- **sybase**

b. **Dialect**: Enter the Hibernate dialect for the server, depending on the server type. The common settings are:

- **org.hibernate.dialect.MySQL5InnoDBDialect**
- **org.hibernate.dialect.MySQL8Dialect**
- **org.hibernate.dialect.MariaDB102Dialect**
- **org.hibernate.dialect.PostgreSQL95Dialect**
- **org.hibernate.dialect.PostgresPlusDialect** (used for EntrepriseDB Postgres Advanced Server)
- **org.hibernate.dialect.SQLServer2012Dialect** (used for MS SQL)
- **org.hibernate.dialect.DB2Dialect**
- **org.hibernate.dialect.Oracle10gDialect**
- **org.hibernate.dialect.SybaseASE15Dialect**
  For a complete list of supported dialects, see Table A.7 in Hibernate properties in the Red Hat JBoss EAP documentation.

c. **Host**: Enter the host name of the external database server.

d. **Port**: Enter the port number of the external database server.

e. **Jdbc URL**: Enter the JDBC URL for the external database server.

> NOTE
>
> If you are using the EntrepriseDB Postgres database server, use an URL starting with **jdbc:postgresql:**// and not with **jdbc:edb:**//. Alternatively, do not set the URL and set the host and port parameters instead.

f. **NonXA**: Select this box if you want to configure the data source in non-XA mode.

g. **JNDI name**: Enter the JNDI name that the application uses for the data source.

h. **User name** and **Password**: Enter the user name and password for the external database server.

i. **Background validation**: Optionally, select this box to enable background SQL validation and enter the background validation interval.

j. Optionally, set the minimum and maximum connection pool sizes, valid connection checker class, and exception sorter class for the database server.

18. If you are using a MySQL version 8 external database server, enable the **mysql_native_password** plugin and use it for authentication. For instructions about this pluding, see Native Pluggable Authentication in the *MySQL 8.0 Reference Manual*.
    If you are using a MySQL version 8 image provided by Red Hat on Red Hat OpenShift Container Platform, to enable the plugin, set the **MYSQL_DEFAULT_AUTHENTICATION_PLUGIN** environment variable to **mysql_native_password**.

    If you created users on the MySQL version 8 server before enabling the **mysql_native_password** plugin, you must update the **mysql-user** table after you enable the plugin.

19. Optionally, depending on your needs, set environment variables. To set an environment variable, click **Add new Environment variable**, then enter the name and value for the variable in the **Name** and **Value** fields.

    - If you want to configure an immutable KIE server that pulls services from the configured Maven repository, enter the following settings:

        i. Set the **KIE_SERVER_CONTAINER_DEPLOYMENT** environment variable. The variable must contain the identifying information of the services (KJAR files) that the deployment must pull from the Maven repository. The format is **<containerId>=<groupId>:<artifactId>:<version>** or, if you want to specify an alias name for the container, **<containerId>(<aliasId>)=<groupId>:<artifactId>:<version>**. You can provide two or more KJAR files using the **|** separator, as illustrated in the following example: **containerId=groupId:artifactId:version|c2(alias2)=g2:a2:v2**.

        ii. Configure an external Maven repository.

    - If you want to configure an external Maven repository, set the following variables:

        - **MAVEN_REPO_URL**: The URL for the Maven repository

        - **MAVEN_REPO_ID**: An identifier for the Maven repository, for example, **repo-custom**

        - **MAVEN_REPO_USERNAME**: The user name for the Maven repository

        - **MAVEN_REPO_PASSWORD**: The password for the Maven repository

    - If your OpenShift environment does not have a connection to the public Internet, configure access to a Maven mirror that you set up according to Section 3.11, "Preparing a Maven mirror repository for offline use". Set the following variables:

        - **MAVEN_MIRROR_URL**: The URL for the Maven mirror repository that you set up in Section 3.11, "Preparing a Maven mirror repository for offline use" . This URL must be accessible from a pod in your OpenShift environment. If you configured this KIE Server as S2I, you already entered this URL.

        - **MAVEN_MIRROR_OF**: The value that determines which artifacts are to be retrieved from the mirror. If you configured this KIE Server as S2I, do not set this value. For instructions about setting the **mirrorOf** value, see Mirror Settings in the Apache Maven documentation. The default value is **external:***. With this value, Maven retrieves every required artifact from the mirror and does not query any other repositories.
          If you configure an external Maven repository (**MAVEN_REPO_URL**), change **MAVEN_MIRROR_OF** to exclude the artifacts in this repository from the mirror, for example, **external:*,!repo-custom**. Replace **repo-custom** with the ID that you configured in **MAVEN_REPO_ID**.

If your authoring environment uses a built-in Business Central Maven repository, change **MAVEN_MIRROR_OF** to exclude the artifacts in this repository from the mirror: **external:*,!repo-rhpamcentr**.

- If you want to configure your KIE Server deployment to use Prometheus to collect and store metrics, set the **PROMETHEUS_SERVER_EXT_DISABLED** environment variable to **false**. For instructions about configuring Prometheus metrics collection, see {URL_MANAGING_SETTINGS}#prometheus-monitoring-ocp-proc_execution-server[*Managing and monitoring KIE Server*].

- If you are using {RH-SSO} authentication and the interaction of your application with {RH-SSO} requires support for CORS, set the **SSO_ENABLE_CORS** variable to **true**.

- In some authoring environments, you might need to ensure that several users can deploy services on the same KIE Server at the same time. By default, after deploying a service onto a KIE Server using Business Central, the user needs to wait for some seconds before more services can be deployed. The **OpenShiftStartupStrategy** setting is enabled by default and causes this limitation. To remove the limitation, you can configure an **rhpam-authoring** environment to use the *controller strategy*. Do not make this change unless a specific need for it exists; if you decide to enable controller strategy, make this change on Business Central and on all KIE Servers in the same environment.

  NOTE

  Do not enable the controller strategy in an environment with a high-availability Business Central. In such environments the controller strategy does not function correctly.

  To enable controller strategy on a KIE Server, set the **KIE_SERVER_STARTUP_STRATEGY** environment variable to **ControllerBasedStartupStrategy** and the **KIE_SERVER_CONTROLLER_OPENSHIFT_ENABLED** environment variable to **false**.

Next steps

To configure additional KIE Servers, click **Add new KIE Server** again and repeat the procedure for the new server configuration.

If you want to deploy the environment without Smart Router and without Process Instance Migration, click **Finish** and then click **Deploy** to deploy the environment. Otherwise, continue to set configuration parameters for Smart Router.

4.2.6. Setting Smart Router configuration for the environment

By default, the deployed environment does not include Smart Router. You can add a Smart Router to the environment. You can also set configuration options for the Smart Router.

Prerequisites

- You completed basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator in the installer wizard according to the instructions in Section 4.2.2, “Setting the basic configuration of the environment”.

Procedure

1. If the **Installation**, **Security**, **Console**, or **KIE Servers** tab is open, click **Next** until you view the **Smart Router** tab.

2. Click **Set Smart Router** to add Smart Router to the environment and to configure Smart Router.

3. If you created the secret for Smart Router according to the instructions in Section 3.5, “Creating the secrets for Smart Router”, enter the name of the secret in the **Secret** field.

4. Optionally, enter the number of replicas for the Smart Router in the **Replicas** field.

5. Optionally, enter requested and maximum CPU and memory limits in the fields under **Resource quotas**.

Next steps

If you want to deploy the Process Instance Migration service, continue to deploy the service. Otherwise, click **Finish** and then click **Deploy** to deploy the environment.

4.2.7. Setting Process Instance Migration configuration for the environment

You can use the operator to deploy the Process Instance Migration (PIM) service. You can use the PIM service to define the migration between two different process definitions, known as a migration plan. You can apply the migration plan to the running process instances in a specific KIE Server.

The PIM service uses a database server for its operation.

Prerequisites

- You completed basic configuration of a Red Hat Process Automation Manager environment using the Business Automation operator in the installer wizard according to the instructions in Section 4.2.2, “Setting the basic configuration of the environment”.

Procedure

1. If the **Installation**, **Security**, **Console**, **KIE Servers**, or **Smart Router** tab is open, click **Next** until you view the **Process Instance Migration** tab.

2. Click **Set Process Instance Migration** to add PIM to the environment and to configure PIM.

3. In the **Database type** field, select the database that the PIM service must use. The following values are available:

   - **mysql**: A MySQL server, created in a separate pod.

   - **postgresql**: A PostgreSQL server, created in a separate pod. Use this setting unless you have a specific reason to use any other setting.

   - **h2**: A built-in **h2** database engine that does not require a separate pod.

4. Optionally, set configuration parameters of the persistent volume for the database:

   - In the **Size** field, enter the size of the persistence volume

   - In the **StorageClass name** field, enter the storage class name for the persistent volume

Next steps

Click **Finish** and then click **Deploy** to deploy the environment.

For instructions about using the PIM service, see Process Instance Migration in *Managing and monitoring business processes in Business Central*.

4.3. MODIFYING AN ENVIRONMENT THAT IS DEPLOYED USING OPERATORS

If an environment is deployed using operators, you cannot modify it using typical OpenShift methods. For example, if you delete a deployment configuration or a service, it is re-created automatically with the same parameters.

To modify the environment, you must modify the YAML description of the environment. You can change common settings such as passwords, add new KIE Servers, and scale KIE Servers.

Procedure

1. Enter your project in the OpenShift web cluster console.
2. In the OpenShift Web console navigation panel, select **Catalog → Installed operators** or **Operators → Installed operators**.
3. Find the **Business Automation** operator line in the table and click **KieApp** in the line. Information about the environments that you deployed using this operator is displayed.
4. Click the name of a deployed environment.
5. Select the **YAML** tab.
   A YAML source is displayed. In this YAML source, you can edit the content under **spec:** to change the configuration of the environment.
6. If you want to change the deployed version of Red Hat Process Automation Manager, add the following line under **spec:**

   ```
   version: 7.8.0
   ```

   You can replace **7.8.0** with another required version. Use this setting to upgrade Red Hat Process Automation Manager to a new version if automatic updates are disabled, for example, if you use a custom image.
7. If you want to change common settings, such as passwords, edit the values under **commonConfig:**.
8. If you want to add new KIE Servers, add their descriptions at the end of the block under **servers:**, as shown in the following examples:
   - To add two servers named **server-a** and **server-a-2**, add the following lines:

     ```
     - deployments: 2
       name: server-a
     ```

   - To add an immutable KIE Server that includes services built from source in an S2I process, add the following lines:

     ```
     - build:
     ```

```
kieServerContainerDeployment: <deployment>
gitSource:
  uri: <url>
  reference: <branch>
  contextDir: <directory>
```

Replace the following values:

- **<deployment>**: The identifying information of the decision service (KJAR file) that is built from your source. The format is **<containerId>=<groupId>:<artifactId>:<version>**. You can provide two or more KJAR files using the **|** separator, for example **containerId=groupId:artifactId:version|c2=g2:a2:v2**. The Maven build process must produce all these files from the source in the Git repository.
- **<url>**: The URL for the Git repository that contains the source for your decision service.
- **<branch>**: The branch in the Git repository.
- **<directory>**: The path to the source within the project downloaded from the Git repository.

9. If you want to scale a KIE Server, find the description of the server in the block under **servers:** and add a **replicas:** setting under that description. For example, **replicas: 3** scales the server to three pods.
10. If you want to make other changes, review the CRD source for the available settings. To view the CRD source, log in to the Red Hat OpenShift Container Platform environment with the **oc** command as an administrative user and then enter the following command:

```
oc get crd kieapps.app.kiegroup.org -o yaml
```

11. Click **Save** and then wait for a **has been updated** pop-up message.
12. Click **Reload** to view the new YAML description of the environment.

4.4. JVM CONFIGURATION PARAMETERS

When deploying Red Hat Process Automation Manager using the operator, you can optionally set a number of JVM configuration parameters for Business Central and KIE Servers. These parameters set environment variables for the corresponding containers.

The following table lists all JVM configuration parameters that you can set when deploying Red Hat Process Automation Manager using the operator.

The default settings are optimal for most use cases. Make any changes only when they are required.

Table 4.1. JVM configuration parameters

Configuration field	Environment variable	Description	Example
Java Opts append	JAVA_OPTS_APPEND	User specified Java options to be appended to generated options in JAVA_OPTS.	**-Dsome.property=foo**

Configuration field	Environment variable	Description	Example
Java max memory ratio	JAVA_MAX_MEM_RATIO	The maximum percentage of container memory that can be used for the Java Virtual Machine. The remaining memory is used for the operating system. The default value is **50**, for a limit of 50%. Sets the **-Xmx** JVM option. If you enter a value of **0**, the **-Xmx** option is not set.	**40**
Java initial memory ratio	JAVA_INITIAL_MEM_RATIO	The percentage of container memory that is initially used for the Java Virtual Machine. The default value is **25**, so 25% of the pod memory is initially allocated for the JVM if this value does not exceed the **Java Max Initial Memory** value. Sets the **-Xms** JVM option. If you enter a value of **0**, the **-Xms** option is not set.	**25**
Java max initial memory	JAVA_MAX_INITIAL_MEM	The maximum amount of memory, in megabytes, that can be initially used for the Java Virtual Machine. If the initial allocated memory, as set in the **Java initial memory ratio** parameter, would otherwise be greater than this value, the amount of memory set in this value is allocated using the **-Xms** JVM option. The default value is **4096**.	**4096**
Java diagnostics	JAVA_DIAGNOSTICS	Enable this setting to enable output of additional JVM diagnostic information to the standard output. Disabled by default.	**true**
Java debug	JAVA_DEBUG	Enable this setting to switch on remote debugging. Disabled by default. Adds the **-agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=${debug_port}** JVM option, where **${debug_port}** defaults to **5005**.	**true**
Java debug port	JAVA_DEBUG_PORT	The port that is used for remote debugging. The default value is **5005**.	**8787**
GC min heap free ratio	GC_MIN_HEAP_FREE_RATIO	Minimum percentage of heap free after garbage collection (GC) to avoid expansion. Sets the **-XX:MinHeapFreeRatio** JVM option.	**20**

Configuration field	Environment variable	Description	Example
GC max heap free ratio	GC_MAX_HEAP_FREE_RATIO	Maximum percentage of heap free after GC to avoid shrinking. Sets the **-XX:MaxHeapFreeRatio** JVM option.	**40**
GC time ratio	GC_TIME_RATIO	Specifies the ratio of the time spent outside the garbage collection (for example, the time spent for application execution) to the time spent in the garbage collection. Sets the **-XX:GCTimeRatio** JVM option.	**4**
GC adaptive size policy weight	GC_ADAPTIVE_SIZE_POLICY_WEIGHT	The weighting given to the current GC time versus previous GC times. Sets the **-XX:AdaptiveSizePolicyWeight** JVM option.	**90**
GC max metaspace size	GC_MAX_METASPACE_SIZE	The maximum metaspace size. Sets the **-XX:MaxMetaspaceSize** JVM option.	**100**

4.5. CREATING CUSTOM IMAGES FOR KIE SERVER

You can create custom images to add files to KIE Server deployments. You must push the images to your own container registry. When deploying Red Hat Process Automation Manager, you can configure the operator to use the custom images.

If you use a custom image, you must disable automatic version updates. When you want to install a new version, build the image with the same name as before and the new version tag and push the image into your registry. You can then change the version and the operator automatically pulls the new image. For instructions about changing the product version in the operator, see Section 4.3, “Modifying an environment that is deployed using operators”.

In particular, you can create the following types of custom images:

- A custom image of KIE Server that includes an additional RPM package
- A custom image of KIE Server that includes an additional JAR library

4.5.1. Creating a custom KIE Server image with an additional RPM package

You can create a custom KIE Server image where an additional RPM package is installed. You can push this image into your custom registry and then use it to deploy the KIE Server.

You can install any package from the Red Hat Enterprise Linux 8 repository. This example installs the **procps-ng** package, which provides the **ps** utility, but you can modify it to install other packages.

Procedure

1. Authenticate to the **registry.redhat.io** registry using the **podman login** command. For instructions about authenticating to the registry, see Red Hat Container Registry Authentication.

2. To download the supported KIE Server base image, enter the following command:

   ```
   podman pull registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
   ```

3. Create a **Dockerfile** that defines a custom image based on the base image. The file must change the current user to **root**, install the RPM package using the **yum** command, and then revert to **USER 185**, the Red Hat JBoss EAP user. The following example shows the content of the **Dockerfile** file:

   ```
   FROM registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
   USER root
   RUN yum -y install procps-ng
   USER 185
   ```

   Replace the name of the RPM file as necessary. The **yum** command automatically installs all dependencies from the Red Hat Enterprise Linux 8 repository. You might need to install several RPM files, in this case, use several **RUN** commands.

4. Build the custom image using the **Dockerfile**. Supply the fully qualified name for the image, including the registry name. You must use the same version tag as the version of the base image. To build the image, enter the following command:

   ```
   podman build . --tag registry_address/image_name:7.8.0
   ```

   For example:

   ```
   podman build . --tag registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
   ```

5. After the build completes, run the image, log in to it, and verify that the customization was successful. Enter the following command:

   ```
   podman run -it --rm registry_address/image_name:7.8.0 /bin/bash
   ```

   For example:

   ```
   podman run -it --rm registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0 /bin/bash
   ```

   In the shell prompt for the image, enter the command to test that the RPM is installed, then enter **exit**. For example, for **procps-ng**, run the **ps** command:

   ```
   [jboss@c2fab36b778e ~]$ ps
   PID TTY          TIME CMD
     1 pts/0    00:00:00 bash
    13 pts/0    00:00:00 ps
   [jboss@c2fab36b778e ~]$ exit
   ```

6. To push the custom image into your registry, enter the following command:

```
podman push registry_address/image_name:7.8.0
docker://registry_address/image_name:7.8.0
```

For example:

```
podman push registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
docker://registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
```

Next steps

When deploying the KIE Server, set the image name and namespace to specify the custom image in your registry. Click **Set KIE Server image**, change the **Kind** value to **DockerImage**, and then provide the image name including the registry name, but without the version tag, for example:

```
registry.example.com/custom/rhpam-kieserver-rhel8
```

For instructions about deploying the KIE Server using the operator, see Section 4.2.5, "Setting custom KIE Server configuration of the environment".

4.5.2. Creating a custom KIE Server image with an additional JAR file

You can create a custom KIE Server image where an additional JAR file (or several JAR files) is installed to extend the capabilities of the server. You can push this image into your custom registry and then use it to deploy the KIE Server.

Procedure

1. Develop a custom library that works with the KIE Server. You can use the following documentation and examples to develop the library:
   - KIE Server capabilities and extensions in *Managing and monitoring KIE Server*.
   - Domain-specific Prometheus metrics with Red Hat Process Automation Manager and Decision Manager
   - Extend KIE Server with additional transport
2. Build the library using Maven, so that the JAR file is placed in the **target** directory. This example uses the **custom-kieserver-ext-1.0.0.Final.jar** file name.
3. Authenticate to the **registry.redhat.io** registry using the **podman login** command. For instructions about authenticating to the registry, see Red Hat Container Registry Authentication.
4. To download the supported KIE Server base image, enter the following command:

   ```
   podman pull registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
   ```

5. Create a **Dockerfile** that defines a custom image based on the base image. The file must copy the JAR file (or several JAR files) into the **/opt/eap/standalone/deployments/ROOT.war/WEB-INF/lib/** directory. The following example shows the content of the **Dockerfile** file:

```
FROM registry.redhat.io/rhpam-7/rhpam-kieserver-rhel8:7.8.0
COPY target/custom-kieserver-ext-1.0.0.Final.jar
/opt/eap/standalone/deployments/ROOT.war/WEB-INF/lib/
```

6. Build the custom image using the **Dockerfile**. Supply the fully qualified name for the image, including the registry name. You must use the same version tag as the version of the base image. To build the image, enter the following command:

   ```
   podman build . --tag registry_address/image_name:7.8.0
   ```

   For example:

   ```
   podman build . --tag registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
   ```

7. To push the custom image into your registry, enter the following command:

   ```
   podman push registry_address/image_name:7.8.0
   docker://registry_address/image_name:7.8.0
   ```

   For example:

   ```
   podman push registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
   docker://registry.example.com/custom/rhpam-kieserver-rhel8:7.8.0
   ```

Next steps

When deploying the KIE Server, set the image name and namespace to specify the custom image in your registry. Click **Set KIE Server image**, change the **Kind** value to **DockerImage**, and then provide the image name including the registry name, but without the version tag, for example:

```
registry.example.com/custom/rhpam-kieserver-rhel8
```

For instructions about deploying the KIE Server using the operator, see Section 4.2.5, “Setting custom KIE Server configuration of the environment”.

CHAPTER 5. MIGRATING INFORMATION FROM A DEPLOYMENT ON RED HAT OPENSHIFT CONTAINER PLATFORM VERSION 3

If you previously used a Red Hat Process Automation Manager deployment on Red Hat OpenShift Container Platform version 3, you can migrate the information from that deployment to a new deployment on Red Hat OpenShift Container Platform version 4.

Before migrating information, you must deploy a new Red Hat Process Automation Manager infrastructure on Red Hat OpenShift Container Platform version 4 using the operator. Include the same elements in the new infrastructure as those present in the old deployment. For example:

- For any existing authoring deployment, create a new authoring infrastructure, including Business Central and at least one KIE Server.
- For any existing immutable KIE Server, deploy a new immutable KIE Server with the same artifacts.
- For any existing KIE Server with a MySQL or PostgreSQL database pod, deploy a new KIE Server with the same type of database pod.
- For any existing KIE Server that uses an external database server, deploy a new KIE Server that uses the same external database server with the same credentials. The server connects to the same database and therefore can read the process context state.

NOTE

If a KIE Server uses the H2 built-in database, migration of the process context state is not supported.

No migration is required for Smart Router. A new deployment of Smart Router automatically works with the services on the new KIE Servers.

5.1. MIGRATING INFORMATION IN BUSINESS CENTRAL

If you have an existing authoring environment in Red Hat OpenShift Container Platform version 3, you can copy the **.niogit** repository and the Maven repository from Business Central in this environment to Business Central in a new deployment on Red Hat OpenShift Container Platform version 4. This action makes all the same projects and artifacts available in the new deployment.

Prerequisites

- You must have a machine that has network access to both the Red Hat OpenShift Container Platform version 3 and Red Hat OpenShift Container Platform version 4 infrastructures.
- The **oc** command-line client from Red Hat OpenShift Container Platform version 4 must be installed on the machine. For instructions about installing the command-line client, see *CLI tools* in Red Hat OpenShift Container Platform documentation.

Procedure

1. Ensure that no web clients and no client applications are connected to any elements of the old and new deployment, including Business Central and KIE Servers.

2. Create an empty temporary directory and change into it.

3. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 3 infrastructure and switch to the project containing the old deployment.

4. To view the pod names in the old deployment, run the following command:

   ```
   oc get pods
   ```

   Find the Business Central pod. The name of this pod includes **rhpamcentr**. In a high-availability deployment, you can use any of the Business Central pods.

5. Use the **oc** command to copy the the **.niogit** repository and the Maven repository from the pod to the local machine, for example:

   ```
   oc cp myapp-rhpamcentr-5-689mw:/opt/kie/data/.niogit .niogit
   oc cp myapp-rhpamcentr-5-689mw:/opt/kie/data/maven-repository maven-repository
   ```

6. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 4 infrastructure and switch to the project containing the new deployment.

7. To view the pod names in the new deployment, run the following command:

   ```
   oc get pods
   ```

   Find the Business Central pod. The name of this pod includes **rhpamcentr**. In a high-availability deployment, you can use any of the Business Central pods.

8. Use the **oc** command to copy the the **.niogit** repository and the Maven repository from the local machine to the pod, for example:

   ```
   oc cp .niogit myappnew-rhpamcentr-abd24:/opt/kie/data/.niogit
   oc cp maven-repository myappnew-rhpamcentr-abd24:/opt/kie/data/maven-repository
   ```

5.2. MIGRATING A MYSQL DATABASE FOR A KIE SERVER

If your environment in Red Hat OpenShift Container Platform version 3 includes a KIE Server that uses a MySQL database pod, copy the MySQL database content from the old deployment to the new deployment. This action copies the existing process state to the new deployment.

Prerequisites

- You must have a machine that has network access to both the Red Hat OpenShift Container Platform version 3 and Red Hat OpenShift Container Platform version 4 infrastructures.

- The **oc** command-line client from Red Hat OpenShift Container Platform version 4 must be installed on the machine. For instructions about installing the command-line client, see *CLI tools* in Red Hat OpenShift Container Platform documentation.

- The **mysql** and **mysqldump** client applications provided by MySQL version 8 or later or by MariaDB version 10 or later must be installed.

Procedure

1. Ensure that no web clients and no client applications are connected to any elements of the old and new deployment, including Business Central and KIE Servers.

2. Create an empty temporary directory and change into it.

3. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 3 infrastructure and switch to the project containing the old deployment.

4. To view the deployment configuration names in the old deployment, run the following command:

   ```
   oc get dc
   ```

   Find the **mysql** deployment configuration that corresponds to the KIE Server.

5. View the configuration YAML of the deployment configuration, for example:

   ```
   oc edit dc/myapp-mysql
   ```

   In this file, find the user name (normally **rhpam**) and password for the database server, for example:

   ```
   - name: MYSQL_USER
     value: rhpam
   - name: MYSQL_PASSWORD
     value: NDaJIV7!
   ```

   Record the user name and password. Do not make any changes to the file.

   NOTE

   You can also use the following commands to retrieve the user name and password:

   ```
   oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_USER")]}'.value

   oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?(@.name=="MYSQL_PASSWORD")]}'.value
   ```

6. To view the service names in the old deployment, run the following command:

   ```
   oc get svc
   ```

   Find the **mysql** service that corresponds to the KIE Server.

7. In a separate terminal window, start port forwarding from the local host to the **mysql** service, using the name and port number displayed for the service, for example:

   ```
   oc port-forward service/myapp-mysql 3306:3306
   ```

8. Create a full database dump using the recorded user name, for example:

```
mysqldump --all-databases -u rhpam -p -h 127.0.0.1 > mysqldump.sql
```

When prompted, enter the recorded password. The dump creation can take considerable time.

9. Stop the port forwarding in the separate window using the kbd:[Ctrl+C] key combination.

10. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 4 infrastructure and switch to the project containing the new deployment.

11. To view the deployment configuration names in the new deployment, run the following command:

    ```
    oc get dc
    ```

    Find the **mysql** deployment configuration that corresponds to the KIE Server.

12. View the configuration YAML of the deployment configuration, for example:

    ```
    oc edit dc/myappnew-mysql
    ```

    In this file, find the user name (normally **rhpam**) and password for the database server. Record the user name and password. Do not make any changes to the file.

    NOTE

    You can also use the following commands to retrieve the user name and password:

    ```
    oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?
    (@.name=="MYSQL_USER")]}'.value

    oc get dc/myapp-mysql -ojsonpath='{.spec.template.spec.containers[0].env[?
    (@.name=="MYSQL_PASSWORD")]}'.value
    ```

13. To view the service names in the new deployment, run the following command:

    ```
    oc get svc
    ```

    Find the **mysql** service that corresponds to the KIE Server.

14. In a separate terminal window, start port forwarding from the local host to the **mysql** service, using the name and port number displayed for the service, for example:

    ```
    oc port-forward service/myappnew-mysql 3306:3306
    ```

15. Restore the database dump using the recorded user name, for example:

    ```
    mysql -u rhpam -p -h 127.0.0.1 < mysqldump.sql
    ```

    When prompted, enter the recorded password. The restoration can take considerable time.

16. Stop the port forwarding in the separate window using the kbd:[Ctrl+C] key combination.

5.3. MIGRATING A POSTGRESQL DATABASE FOR A KIE SERVER

If your environment in Red Hat OpenShift Container Platform version 3 includes a KIE Server that uses a PostgreSQL database pod, copy the PostgreSQL database content from the old deployment to the new deployment. This action copies the existing process state to the new deployment.

Prerequisites

- You must have a machine that has network access to both the Red Hat OpenShift Container Platform version 3 and Red Hat OpenShift Container Platform version 4 infrastructures.
- The **oc** command-line client from Red Hat OpenShift Container Platform version 4 must be installed on the machine. For instructions about installing the command-line client, see *CLI tools* in Red Hat OpenShift Container Platform documentation.
- The **psql** and **pg_dump** client applications provided by PostgreSQL version 10 or later must be installed.

Procedure

1. Ensure that no web clients and no client applications are connected to any elements of the old and new deployment, including Business Central and KIE Servers.
2. Create an empty temporary directory and change into it.
3. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 3 infrastructure and switch to the project containing the old deployment.
4. To view the deployment configuration names in the old deployment, run the following command:

   ```
   oc get dc
   ```

   Find the **postgresql** deployment configuration that corresponds to the KIE Server.
5. View the configuration YAML of the deployment configuration, for example:

   ```
   oc edit dc/myapp-postgresql
   ```

   In this file, find the user name (normally **rhpam**), password, and database name (normally **rhpam7**) for the database server, for example:

   ```
   - name: POSTGRESQL_USER
     value: rhpam
   - name: POSTGRESQL_PASSWORD
     value: NDaJIV7!
   - name: POSTGRESQL_DATABASE
     value: rhpam7
   ```

   Record the user name, password, and database name. Do not make any changes to the file.

NOTE

You can also use the following commands to retrieve the user name, password, and database name:

```
oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_USER")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_PASSWORD")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_DATABASE")]}'.value
```

+

6. To view the service names in the old deployment, run the following command:

   ```
   oc get svc
   ```

   Find the **postgresql** service that corresponds to the KIE Server.

7. In a separate terminal window, start port forwarding from the local host to the **postgresql** service, using the name and port number displayed for the service, for example:

   ```
   oc port-forward service/myapp-postgresql 5432:5432
   ```

8. Create a dump of the database using the recorded user name and database name, for example:

   ```
   pg_dump rhpam7 -h 127.0.0.1 -U rhpam -W > pgdump.sql
   ```

   When prompted, enter the recorded password. The dump creation can take considerable time.

9. Stop the port forwarding in the separate window using the kbd:[Ctrl+C] key combination.

10. Using the **oc** command, log in to the Red Hat OpenShift Container Platform version 4 infrastructure and switch to the project containing the new deployment.

11. To view the deployment configuration names in the new deployment, run the following command:

    ```
    oc get dc
    ```

    Find the **postgresql** deployment configuration that corresponds to the KIE Server.

12. View the configuration YAML of the deployment configuration, for example:

    ```
    oc edit dc/myappnew-postgresql
    ```

In this file, find the user name (normally **rhpam**), password, , and database name (normally **rhpam7**) for the database server. Record the user name, password, and database name. Do not make any changes to the file.

NOTE

You can also use the following commands to retrieve the user name, password, and database name:

```
oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_USER")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_PASSWORD")]}'.value

oc get dc/myapp-postgresql -
ojsonpath='{.spec.template.spec.containers[0].env[?
(@.name=="POSTGRESQL_DATABASE")]}'.value
```

13. To view the service names in the new deployment, run the command:

    ```
    oc get svc
    ```

    Find the **postgresql** service that corresponds to the KIE Server.

14. In a separate terminal window, start port forwarding from the local host to the **postgresql** service, using the name and port number displayed for the service, for example:

    ```
    oc port-forward service/myappnew-postgresql 5432:5432
    ```

15. Restore the database dump using the recorded user name and database name, for example:

    ```
    psql -h 127.0.0.1 -d rhpam7 -U rhpam -W < pgdump.sql
    ```

    When prompted, enter the recorded password. The restoration can take considerable time.

    Review any displayed database error messages. Messages about objects that already exist are normal.

16. Stop the port forwarding in the separate window using the kbd:[Ctrl+C] key combination.

APPENDIX A. VERSIONING INFORMATION

Documentation last updated on Tuedday, March 8, 2022.