Red Hat Decision Manager 7.2

Deploying a Red Hat Decision Manager immutable server environment on Red Hat OpenShift Container Platform

Last Updated: 2020-05-04

Red Hat Decision Manager 7.2 Deploying a Red Hat Decision Manager immutable server environment on Red Hat OpenShift Container Platform

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

Legal Notice

Abstract

This document describes how to deploy a Red Hat Decision Manager 7.2 immutable server environment on Red Hat OpenShift Container Platform.

Table of Contents

PREFACE 3

CHAPTER 1. OVERVIEW OF RED HAT DECISION MANAGER ON RED HAT OPENSHIFT CONTAINER PLATFORM 4

CHAPTER 2. PREPARING TO DEPLOY RED HAT DECISION MANAGER IN YOUR OPENSHIFT ENVIRONMENT 6

2.1. ENSURING THE AVAILABILITY OF IMAGE STREAMS AND THE IMAGE REGISTRY 6
2.2. CREATING THE SECRETS FOR DECISION SERVER 7
2.3. EXTRACTING THE SOURCE CODE FROM DECISION CENTRAL FOR USE IN AN S2I BUILD 8
2.4. PREPARING A MAVEN REPOSITORY FOR OFFLINE USE 8

CHAPTER 3. ENVIRONMENT WITH IMMUTABLE SERVERS 9

3.1. DEPLOYING AN IMMUTABLE DECISION SERVER FROM SERVICE SOURCE CODE 9
3.2. DEPLOYING AN IMMUTABLE DECISION SERVER FROM KJAR SERVICES 12
3.3. PROVIDING THE LDAP ROLE MAPPING FILE 15

CHAPTER 4. OPENSHIFT TEMPLATE REFERENCE INFORMATION 17

4.1. RHDM72-PROD-IMMUTABLE-KIESERVER.YAML TEMPLATE 17
4.1.1. Parameters 17
4.1.2. Objects 31
4.1.2.1. Services 31
4.1.2.2. Routes 31
4.1.2.3. Build Configurations 31
4.1.2.4. Deployment Configurations 31
4.1.2.4.1. Triggers 32
4.1.2.4.2. Replicas 32
4.1.2.4.3. Pod Template 32
4.1.2.4.3.1. Service Accounts 32
4.1.2.4.3.2. Image 32
4.1.2.4.3.3. Readiness Probe 32
4.1.2.4.3.4. Liveness Probe 33
4.1.2.4.3.5. Exposed Ports 33
4.1.2.4.3.6. Image Environment Variables 33
4.1.2.4.3.7. Volumes 42
4.1.2.5. External Dependencies 42
4.1.2.5.1. Secrets 43
4.2. OPENSHIFT USAGE QUICK REFERENCE 43

APPENDIX A. VERSIONING INFORMATION 45

PREFACE

As a system engineer, you can deploy a Red Hat Decision Manager immutable server environment on Red Hat OpenShift Container Platform to provide an infrastructure to execute services and other business assets. You can use standard integration tools to manage the immutable Decision Server image. You can create new server images to add and update the business assets.

Prerequisites

- At least two gigabytes of memory must be available in the OpenShift cluster/namespace.
- The OpenShift project for the deployment must be created.
- You must be logged in to the project using the **oc** command. For more information about the **oc** command-line tool, see the OpenShift CLI Reference. If you want to use the OpenShift Web console to deploy templates, you must also be logged on using the Web console.

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

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

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

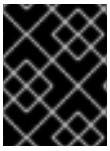
The following key components of Red Hat Decision Manager are available on OpenShift:

- Decision Server, also known as *Execution Server* or *KIE Server*, is the infrastructure element that runs decision services and other deployable assets (collectively referred to as *services*) . All logic of the services runs on execution servers.
  You can freely scale up a Decision Server pod, providing as many copies as necessary, running on the same host or different hosts. As you scale a pod up or down, all its copies run the same services. OpenShift provides load balancing and a request can be handled by any of the pods.

  You can deploy a separate Decision 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 Decision Server pods as necessary.

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

  Decision 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, Decision Central can place the compiled services (KJAR files) into the built-in Maven repository or (if configured) into an external Maven repository.



IMPORTANT

In the current version, high-availability Decision Central functionality is a technology preview.

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

The following environment types are typical:

- *Authoring or managed environment* : An environment architecture that can be used for creating and modifying services using Decision Central and also for running services on Decision Servers. It consists of pods that provide Decision Central for the authoring work and one or more Decision Servers for execution of the services. Each Decision Server is a pod that you can replicate by scaling it up or down as necessary. You can deploy and undeploy services on each Decision Server using Decision Central. For instructions about deploying this environment, see *Deploying a Red Hat Decision Manager authoring or managed server environment on Red Hat OpenShift Container Platform*.

- *Deployment with immutable servers* : An alternate environment for running existing services for staging and production purposes. In this environment, when you deploy a Decision Server pod, it builds an image that loads and starts a service or group of services. You cannot stop any service

on the pod or add any new service to the pod. If you want to use another version of a service or modify the configuration in any other way, you deploy a new server image and displace the old one. In this system, the Decision Server runs like any other pod on the OpenShift environment; you can use any container-based integration workflows and do not need to use any other tools to manage the pods. For instructions about deploying this environment, see *Deploying a Red Hat Decision Manager immutable server environment on Red Hat OpenShift Container Platform*.

You can also deploy a *trial* or evaluation environment. This environment includes Decision Central and a Decision Server. You can set it up quickly and use it to evaluate or demonstrate developing and running assets. However, the environment does not use any persistent storage, and any work you do in the environment is not saved. For instructions about deploying this environment, see *Deploying a Red Hat Decision Manager trial environment on Red Hat OpenShift Container Platform*.

To deploy a Red Hat Decision Manager environment on OpenShift, you can use the templates that are provided with Red Hat Decision Manager.

CHAPTER 2. PREPARING TO DEPLOY RED HAT DECISION MANAGER IN YOUR OPENSHIFT ENVIRONMENT

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

2.1. ENSURING THE AVAILABILITY OF IMAGE STREAMS AND THE IMAGE REGISTRY

To deploy Red Hat Decision Manager components of Red Hat OpenShift Container Platform, you must ensure that OpenShift can download the correct images from the Red Hat registry. To download the images, OpenShift requires the information about their location (known as *image streams*). OpenShift also must be configured to authenticate with the Red Hat registry using your service account user name and password.

Some versions of the OpenShift environment include the required image streams. You must check if they are available. If image streams are available in OpenShift by default, you can use them if the OpenShift infrastructure is configured for registry authentication server. The administrator must complete the registry authentication configuration when installing the OpenShift environment.

Otherwise, you can configure registry authentication in your own project and install the image streams in the same project.

Procedure

1. Determine whether Red Hat OpenShift Container Platform was configured with the user name and password for Red Hat registry access. For details about the required configuration, see Configuring a Registry Location. If you are using an OpenShift Online subscription, it is configured for Red Hat registry access.

2. If Red Hat OpenShift Container Platform was configured with the user name and password for Red Hat registry access, run the following commands:

```
$ oc get imagestreamtag -n openshift | grep rhdm72-decisioncentral-openshift
$ oc get imagestreamtag -n openshift | grep rhdm72-kieserver-openshift
```

   If the outputs of both commands are not empty, the required image streams are available in the **openshift** namespace and no further action is required.

3. If the output of one or both of the commands is empty or if OpenShift was not configured with the user name and password for Red Hat registry access, complete the following steps:

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

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

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

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

   e. 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
```

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

f. Download the **rhdm-7.2.0-openshift-templates.zip** product deliverable file from the Software Downloads page and extract the **rhdm72-image-streams.yaml** file.

g. Complete one of the following actions:

- Run the following command:

```
$ oc create -f rhdm72-image-streams.yaml
```

- Using the OpenShift Web UI, select **Add to Project → Import YAML / JSON** and then choose the file or paste its contents.

NOTE

If you complete these steps, you install the image streams into the namespace of your project. If you install the image streams using these steps, you must set the **IMAGE_STREAM_NAMESPACE** parameter to the name of this project when deploying templates.

2.2. CREATING THE SECRETS FOR DECISION SERVER

OpenShift uses objects called **Secrets** to hold sensitive information, such as passwords or keystores. See the Secrets chapter in the OpenShift documentation for more information.

You must create an SSL certificate for Decision Server and provide it to your OpenShift environment as a secret.

NOTE

You do not need to create the secrets object if you are planning to deploy only Decision Servers without support for HTTPS.

Procedure

1. Generate an SSL keystore with a private and public key for SSL encryption for Decision Server. In a production environment, generate a valid signed certificate that matches the expected URL of the Decision Server. Save the keystore in a file named **keystore.jks**. Record the name of the certificate and the password of the keystore file.
   See Generate a SSL Encryption Key and Certificate for more information on how to create a keystore with self-signed or purchased SSL certificates.

2. 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
```

2.3. EXTRACTING THE SOURCE CODE FROM DECISION CENTRAL FOR USE IN AN S2I BUILD

If you are using Decision 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.

Procedure

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

   ```
   git clone ssh://adminUser@decision-central-host:8001/MySpace/MyProject
   ```

   Replace:

   - **adminUser** with the administrative user for Decision Central
   - **decision-central-host** with the host on which Decision Central is running
   - **MySpace** with the name of the Decision Central space in which the project is located
   - **MyProject** with the name of the project

2. Upload the source code to another Git repository for the S2I build.

2.4. PREPARING A MAVEN 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 for use in source to image (S2I) builds.

Skip this procedure if your Red Hat OpenShift Container Platform environment is connected to the Internet.

Procedure

1. Build the source of your services on any machine using the **mvn clean install** command.
2. Copy the downloaded Maven artifacts from the machine onto an internal Maven repository (for example, Nexus).
3. Make this repository available in your Red Hat OpenShift Container Platform environment.

CHAPTER 3. ENVIRONMENT WITH IMMUTABLE SERVERS

You can deploy an environment that includes one or more pods running Decision Server with preloaded services. Each Decision Server pod can be separately scaled as necessary.

In this case, any services (KJAR files) must be loaded onto a Decision Server at the time the image is created. You cannot load or unload services on a running Decision Server. The advantage of this approach is that the Decision Server with the services in it runs like any other containerized service and does not require specialized management. The Decision Server runs like any other pod on the OpenShift environment; you can use any container-based integration workflows as necessary.

When you create a Decision Server image, you can build your services using S2I (Source to Image). Provide a Git repository with the source of your services and other business assets; if you develop the services or assets in Decision Central, copy the source into a separate repository for the S2I build. OpenShift automatically builds the source, installs the services into the Decision Server image, and starts the containers with the services.

If you are using Decision Central for authoring services, you can extract the source for your process and place it into a separate Git repository (such as GitHub or an on-premise installation of GitLab) for use in the S2I build.

Alternatively, you can create a similar Decision Server deployment using services that are already built as KJAR files. In this case, you must provide the services in a Maven repository; you can use the built-in repository of the Decision Central or your own repository (for example, a Nexus deployment). The KJAR files are retrieved from the Maven repository during the startup of the pod and not updated or changed after that. The files are retrieved at every restart or scaling of the pod, so you must ensure they do not change on the Maven repository to keep the deployment immutable.

With both methods of creating immutable images, no further management of the image is required. If you want to use a new version of a service, you can build a new image.

3.1. DEPLOYING AN IMMUTABLE DECISION SERVER FROM SERVICE SOURCE CODE

To deploy an immutable Decision Server from service source code, use the **rhdm72-prod-immutable-kieserver.yaml** template file. You can extract this file from the **rhdm-7.2.0-openshift-templates.zip** product deliverable file. You can download the file from the Software Downloads page.

When you deploy an immutable Decision Server, the deployment procedure retrieves the source code for any services that must run on this server, builds the services, and includes them in the server image.

Procedure

1. Use one of the following methods to deploy the template:

   - In the OpenShift Web UI, select **Add to Project → Import YAML / JSON** and then select or paste the **rhdm72-prod-immutable-kieserver.yaml** file. In the **Add Template** window, ensure **Process the template** is selected and click **Continue**.

   - To use the OpenShift command line console, prepare the following command line:

     ```
     oc new-app -f <template-path>/rhdm72-prod-immutable-kieserver.yaml -p
     KIE_SERVER_HTTPS_SECRET=kieserver-app-secret
     ```

     In this command line:

- Replace **<template-path>** with the path to the downloaded template file.
- Use as many **-p PARAMETER=value** pairs as needed to set the required parameters. You can view the template file to see descriptions for all parameters.

2. Set the following parameters as necessary:

- **KIE Server Keystore Secret Name**(**KIE_SERVER_HTTPS_SECRET**): The name of the secret for Decision Server, as created in Section 2.2, "Creating the secrets for Decision Server".
- **Application Name** (**APPLICATION_NAME**): The name of the OpenShift application. It is used in the default URL for Decision Server. OpenShift uses the application name to create a separate set of deployment configurations, services, routes, labels, and artifacts. You can deploy several applications using the same template into the same project, as long as you use different application names. Also, the application name determines the name of the server configuration (server template) on the Decision Central that the Decision Server is to join. If you are deploying several Decision Servers, you must ensure each of the servers has a different application name.
- **KIE Server Certificate Name**(**KIE_SERVER_HTTPS_NAME**): The name of the certificate in the keystore that you created in Section 2.2, "Creating the secrets for Decision Server" .
- **KIE Server Keystore Password** (**KIE_SERVER_HTTPS_PASSWORD**): The password for the keystore that you created in Section 2.2, "Creating the secrets for Decision Server" .
- **KIE Server Container Deployment**(**KIE_SERVER_CONTAINER_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.
- **Git Repository URL** (**SOURCE_REPOSITORY_URL**): The URL for the Git repository that contains the source for your decision service.
- **Git Reference** (**SOURCE_REPOSITORY_REF**): The branch in the Git repository
- **Context Directory** (**CONTEXT_DIR**): The path to the source within the project downloaded from the Git repository
- **Artifact Directory** (**ARTIFACT_DIR**): 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
- **ImageStream Namespace** (**IMAGE_STREAM_NAMESPACE**): The namespace where the image streams are available. If the image streams were already available in your OpenShift environment (see Section 2.1, "Ensuring the availability of image streams and the image registry"), the namespace is **openshift**. If you have installed the image streams file, the namespace is the name of the OpenShift project.

3. If your build includes dependencies that are not available on the public Maven tree and require a separate repository, set the parameters to provide this repository:

- **Maven repository URL**(**MAVEN_REPO_URL**): The URL for the Maven repository.

- **Maven repository username** (**MAVEN_REPO_USERNAME**): The username for the Maven repository.
- **Maven repository password** (**MAVEN_REPO_PASSWORD**): The password for the Maven repository.

4. If you want to use RH-SSO or LDAP authentication, complete the following additional configuration. Do not configure LDAP authentication and RH-SSO authentication in the same deployment.

   a. In the RH-SSO or LDAP service, create all user names in the deployment parameters. If you do not set any of the parameters, create users with the default user names. The created users must also be assigned to roles:

      - **KIE_ADMIN_USER**: default user name **adminUser**, roles: **kie-server,rest-all,admin**
      - **KIE_SERVER_USER**: default user name **executionUser**, roles **kie-server,rest-all,guest**

   b. If you want to configure Red Hat Single Sign On (RH-SSO) authentication, an RH-SSO realm that applies to Red Hat Decision Manager must exist. A client within RH-SSO must also exist for
   For the user roles that you can configure in RH-SSO, see Roles and users.

      Use one of the following procedures:

      i. If the client for Red Hat Decision Manager within RH-SSO already exists, set the following parameters in the template:

         - **RH-SSO URL** (**SSO_URL**): The URL for RH-SSO.
         - **RH-SSO Realm name** (**SSO_REALM**): The RH-SSO realm for Red Hat Decision Manager.
         - **KIE Server RH-SSO Client name** (**KIE_SERVER_SSO_CLIENT**): The RH-SSO client name for Decision Server.
         - **KIE Server RH-SSO Client Secret** (**KIE_SERVER_SSO_SECRET**): The secret string that is set in RH-SSO for the client for Decision Server.
         - **RH-SSO Disable SSL Certificate Validation** (**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**): Set to **true** if your RH-SSO installation does not use a valid HTTPS certificate.

      ii. To create the client for Red Hat Decision Manager within RH-SSO, set the following parameters in the template:

         - **RH-SSO URL** (**SSO_URL**): The URL for RH-SSO.
         - **RH-SSO Realm name** (**SSO_REALM**): The RH-SSO realm for Red Hat Decision Manager.
         - **KIE Server RH-SSO Client name** (**KIE_SERVER_SSO_CLIENT**): The name of the client to create in RH-SSO for Decision Server.
         - **KIE Server RH-SSO Client Secret** (**KIE_SERVER_SSO_SECRET**): The secret string to set in RH-SSO for the client for Decision Server.

- **KIE Server Custom http Route Hostname**(**KIE_SERVER_HOSTNAME_HTTP**): The fully qualified host name to use for the HTTP endpoint for Decision Server. If you need to create a client in RH-SSO, you can not leave this parameter blank.
- **KIE Server Custom https Route Hostname** (**KIE_SERVER_HOSTNAME_HTTPS**): The fully qualified host name to use for the HTTPS endpoint for Decision Server. If you need to create a client in RH-SSO, you can not leave this parameter blank.
- **RH-SSO Realm Admin Username** (**SSO_USERNAME**) and **RH-SSO Realm Admin Password** (**SSO_PASSWORD**): The user name and password for the realm administrator user for the RH-SSO realm for Red Hat Decision Manager.
- **RH-SSO Disable SSL Certificate Validation** (**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**): Set to **true** if your RH-SSO installation does not use a valid HTTPS certificate.

c. To configure LDAP, set the **AUTH_LDAP*** parameters of the template. These parameters correspond to the settings of the LdatExtended Login module of Red Hat JBoss EAP. For instructions about using these settings, see LdapExtended Login Module.
If the LDAP server does not define all the roles required for your deployment, you can map LDAP groups to Red Hat Decision Manager roles. To enable LDAP role mapping, set the following parameters:

- **RoleMapping rolesProperties file path** (**AUTH_ROLE_MAPPER_ROLES_PROPERTIES**): The fully qualified pathname of a file that defines role mapping, for example, **/opt/eap/standalone/configuration/rolemapping/rolemapping.properties**. You must provide this file and mount it at this path in all applicable deployment configurations; for instructions, see Section 3.3, "Providing the LDAP role mapping file".
- **RoleMapping replaceRole property** (**AUTH_ROLE_MAPPER_REPLACE_ROLE**): If set to **true**, mapped roles replace the roles defined on the LDAP server; if set to **false**, both mapped roles and roles defined on the LDAP server are set as user application roles. The default setting is **false**.

5. Complete the creation of the environment, depending on the method that you are using:
   - In the OpenShift Web UI, click **Create**.
     - If the **This will create resources that may have security or project behavior implications** message appears, click **Create Anyway**.
   - Complete and run the command line.

3.2. DEPLOYING AN IMMUTABLE DECISION SERVER FROM KJAR SERVICES

To deploy an immutable Decision Server from KJAR services, use the **rhdm72-kieserver.yaml** template file. You can extract this file

from the **rhdm-7.2.0-openshift-templates.zip** product deliverable file. You can download the file from the Software Downloads page.

In this method of deployment, the Decision Server retrieves all the required KJAR files during the startup of the pod.

Procedure

1. Use one of the following methods to deploy the template:

   - In the OpenShift Web UI, select **Add to Project → Import YAML / JSON** and then select or paste the template file. In the **Add Template** window, ensure **Process the template** is selected and click **Continue**.

   - To use the OpenShift command line console, prepare the following command line:

     ```
     oc new-app -f <template-path>/<template-file-name>.yaml -p
     KIE_SERVER_HTTPS_SECRET=kieserver-app-secret
     ```

     In this command line:

     - Replace **<template-path>** with the path to the template file.
     - Replace **<template-file-name>** with the name of the template file.
     - Use as many **-p PARAMETER=value** pairs as needed to set the required parameters. You can view the template file to see descriptions for all parameters.

2. Set the following parameters as necessary:

   - **KIE Server Keystore Secret Name** (**KIE_SERVER_HTTPS_SECRET**): The name of the secret for Decision Server, as created in Section 2.2, "Creating the secrets for Decision Server".
   - **Application Name** (**APPLICATION_NAME**): The name of the OpenShift application. It is used in the default URL for Decision Server. OpenShift uses the application name to create a separate set of deployment configurations, services, routes, labels, and artifacts. You can deploy several applications using the same template into the same project, as long as you use different application names. Also, the application name determines the name of the server configuration (server template) on the Decision Central that the Decision Server is to join. If you are deploying several Decision Servers, you must ensure each of the servers has a different application name.
   - **KIE Server Certificate Name** (**KIE_SERVER_HTTPS_NAME**): The name of the certificate in the keystore that you created in Section 2.2, "Creating the secrets for Decision Server".
   - **KIE Server Keystore Password** (**KIE_SERVER_HTTPS_PASSWORD**): The password for the keystore that you created in Section 2.2, "Creating the secrets for Decision Server".
   - **KIE Server Container Deployment** (**KIE_SERVER_CONTAINER_DEPLOYMENT**): The identifying information of the decision services (KJAR files) that the deployment must pull from the Maven repository. 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**.
   - **Maven repository URL** (**MAVEN_REPO_URL**): The URL for the Maven repository.
   - **Maven repository username** (**MAVEN_REPO_USERNAME**): The username for the Maven repository.
   - **Maven repository password** (**MAVEN_REPO_PASSWORD**): The password for the Maven repository.

- **Disable KIE server management**(**KIE_SERVER_MGMT_DISABLED**): You must set this parameter to **true** for an immutable deployment.
- **KIE Server Startup Strategy**(**KIE_SERVER_STARTUP_STRATEGY**): You must set this parameter to **LocalContainersStartupStrategy** for an immutable deployment.
- **ImageStream Namespace** (**IMAGE_STREAM_NAMESPACE**): The namespace where the image streams are available. If the image streams were already available in your OpenShift environment (see Section 2.1, "Ensuring the availability of image streams and the image registry"), the namespace is **openshift**. If you have installed the image streams file, the namespace is the name of the OpenShift project.

3. If you want to use RH-SSO or LDAP authentication, complete the following additional configuration. Do not configure LDAP authentication and RH-SSO authentication in the same deployment.

   a. In the RH-SSO or LDAP service, create all user names in the deployment parameters. If you do not set any of the parameters, create users with the default user names. The created users must also be assigned to roles:

      - **KIE_ADMIN_USER**: default user name **adminUser**, roles: **kie-server,rest-all,admin**
      - **KIE_SERVER_USER**: default user name **executionUser**, roles **kie-server,rest-all,guest**

   b. If you want to configure Red Hat Single Sign On (RH-SSO) authentication, an RH-SSO realm that applies to Red Hat Decision Manager must exist. A client within RH-SSO must also exist for
   For the user roles that you can configure in RH-SSO, see Roles and users.

      Use one of the following procedures:

      i. If the client for Red Hat Decision Manager within RH-SSO already exists, set the following parameters in the template:

         - **RH-SSO URL** (**SSO_URL**): The URL for RH-SSO.
         - **RH-SSO Realm name** (**SSO_REALM**): The RH-SSO realm for Red Hat Decision Manager.
         - **KIE Server RH-SSO Client name**(**KIE_SERVER_SSO_CLIENT**): The RH-SSO client name for Decision Server.
         - **KIE Server RH-SSO Client Secret**(**KIE_SERVER_SSO_SECRET**): The secret string that is set in RH-SSO for the client for Decision Server.
         - **RH-SSO Disable SSL Certificate Validation** (**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**): Set to **true** if your RH-SSO installation does not use a valid HTTPS certificate.

      ii. To create the client for Red Hat Decision Manager within RH-SSO, set the following parameters in the template:

         - **RH-SSO URL** (**SSO_URL**): The URL for RH-SSO.
         - **RH-SSO Realm name** (**SSO_REALM**): The RH-SSO realm for Red Hat Decision Manager.

- **KIE Server RH-SSO Client name** (**KIE_SERVER_SSO_CLIENT**): The name of the client to create in RH-SSO for Decision Server.
- **KIE Server RH-SSO Client Secret** (**KIE_SERVER_SSO_SECRET**): The secret string to set in RH-SSO for the client for Decision Server.
- **KIE Server Custom http Route Hostname** (**KIE_SERVER_HOSTNAME_HTTP**): The fully qualified host name to use for the HTTP endpoint for Decision Server. If you need to create a client in RH-SSO, you can not leave this parameter blank.
- **KIE Server Custom https Route Hostname** (**KIE_SERVER_HOSTNAME_HTTPS**): The fully qualified host name to use for the HTTPS endpoint for Decision Server. If you need to create a client in RH-SSO, you can not leave this parameter blank.
- **RH-SSO Realm Admin Username** (**SSO_USERNAME**) and **RH-SSO Realm Admin Password** (**SSO_PASSWORD**): The user name and password for the realm administrator user for the RH-SSO realm for Red Hat Decision Manager.
- **RH-SSO Disable SSL Certificate Validation** (**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**): Set to **true** if your RH-SSO installation does not use a valid HTTPS certificate.

c. To configure LDAP, set the **AUTH_LDAP*** parameters of the template. These parameters correspond to the settings of the LdatExtended Login module of Red Hat JBoss EAP. For instructions about using these settings, see LdapExtended Login Module.
If the LDAP server does not define all the roles required for your deployment, you can map LDAP groups to Red Hat Decision Manager roles. To enable LDAP role mapping, set the following parameters:

- **RoleMapping rolesProperties file path** (**AUTH_ROLE_MAPPER_ROLES_PROPERTIES**): The fully qualified pathname of a file that defines role mapping, for example, **/opt/eap/standalone/configuration/rolemapping/rolemapping.properties**. You must provide this file and mount it at this path in all applicable deployment configurations; for instructions, see Section 3.3, "Providing the LDAP role mapping file".
- **RoleMapping replaceRole property** (**AUTH_ROLE_MAPPER_REPLACE_ROLE**): If set to **true**, mapped roles replace the roles defined on the LDAP server; if set to **false**, both mapped roles and roles defined on the LDAP server are set as user application roles. The default setting is **false**.

4. Complete the creation of the environment, depending on the method that you are using:
   - In the OpenShift Web UI, click **Create**.
     - If the **This will create resources that may have security or project behavior implications** message appears, click **Create Anyway**.
   - Complete and run the command line.

3.3. PROVIDING THE LDAP ROLE MAPPING FILE

If you configure the **AUTH_ROLE_MAPPER_ROLES_PROPERTIES** parameter, you must provide a file that defines the role mapping. Mount this file on all affected deployment configurations.

Procedure

1. Create the role mapping properties file, for example, **my-role-map**. The file must contain entries in the following format:

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

   For example:

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

2. Create an OpenShift configuration map from the file. Run the following command:

```
oc create configmap ldap_role_mapping --from-file=<new_name>=<existing_name>
```

   Where **new_name** is the name that the file is to have on the pods (it must be the same as the name specified in the **AUTH_ROLE_MAPPER_ROLES_PROPERTIES** file) and **existing_name** is the name of the file that you created. For example:

```
oc create configmap ldap_role_mapping --from-file=rolemapping.properties=my-role-map
```

3. Mount the configuration map on every deployment config that is configured for role mapping. The following deployment configs can be affected in this environment:

   - ***myapp*-rhpamcentrmon**: Decision Central Monitoring
   - ***myapp*-kieserver**: Decision Server

   Where **myapp** is the application name. Sometimes, several Decision Server deployments can be present under different application names.

   For every deployment configuration, run the command:

```
oc set volume dc/<deployment_config_name> --add --type configmap --configmap-name
ldap_role_mapping --mount-path=<mapping_dir> --name=ldap_role_mapping
```

   Where **mapping_dir** is the directory name (without file name) set in the **AUTH_ROLE_MAPPER_ROLES_PROPERTIES** parameter, for example, **/opt/eap/standalone/configuration/rolemapping** .

CHAPTER 4. OPENSHIFT TEMPLATE REFERENCE INFORMATION

Red Hat Decision Manager provides the following OpenShift templates. To access the templates, download and extract the **rhdm-7.2.0-openshift-templates.zip** product deliverable file from the Software Downloads page of the Red Hat customer portal.

- **rhdm72-prod-immutable-kieserver.yaml** provides an immutable Decision Server. Deployment of this template includes a source-to-image (S2I) build for one or several services that are to run on the Decision Server. For details about this template, see Section 4.1, "rhdm72-prod-immutable-kieserver.yaml template".

4.1. RHDM72-PROD-IMMUTABLE-KIESERVER.YAML TEMPLATE

Application template for an immutable KIE server in a production environment, for Red Hat Decision Manager 7.2

4.1.1. Parameters

Templates allow you to define parameters which take on a value. That value is then substituted wherever the parameter is referenced. References can be defined in any text field in the objects list field. Refer to the Openshift documentation for more information.

Variable name	Image Environment Variable	Description	Example value	Required
APPLICATION_NAME	–	The name for the application.	myapp	True
KIE_ADMIN_USER	**KIE_ADMIN_USER**	KIE administrator username	adminUser	False
KIE_ADMIN_PWD	**KIE_ADMIN_PWD**	KIE administrator password	–	False
KIE_SERVER_USER	**KIE_SERVER_USER**	KIE server username (Sets the org.kie.server.user system property)	executionUser	False

Variable name	Image Environment Variable	Description	Example value	Required
KIE_SERVER_PWD	**KIE_SERVER_PWD**	KIE server password, used to connect to KIE servers. Generated value can be a suggestion to use for thew s2i various (Sets the org.kie.server.pwd system property)	–	False
IMAGE_STREAM_NAMESPACE	–	Namespace in which the ImageStreams for Red Hat Middleware images are installed. These ImageStreams are normally installed in the openshift namespace. You should only need to modify this if you installed the ImageStreams in a different namespace/project.	openshift	True
KIE_SERVER_IMAGE_STREAM_NAME	–	The name of the image stream to use for KIE server. Default is "rhdm72-kieserver-openshift".	rhdm72-kieserver-openshift	True
IMAGE_STREAM_TAG	–	A named pointer to an image in an image stream. Default is "1.1".	1.1	True

Variable name	Image Environment Variable	Description	Example value	Required
KIE_SERVER_CONTROLLER_USER	**KIE_SERVER_CONTROLLER_USER**	KIE server controller username (Sets the org.kie.server.controller.user system property)	controllerUser	False
KIE_SERVER_CONTROLLER_PWD	**KIE_SERVER_CONTROLLER_PWD**	KIE server controller password (Sets the org.kie.server.controller.pwd system property)	–	False
KIE_SERVER_CONTROLLER_TOKEN	**KIE_SERVER_CONTROLLER_TOKEN**	KIE server controller token for bearer authentication (Sets the org.kie.server.controller.token system property)	–	False
KIE_SERVER_CONTROLLER_SERVICE	**KIE_SERVER_CONTROLLER_SERVICE**	The service name for the optional standalone controller. The application uses this service name to register with the controller. (If set, will be used to discover host and port)	–	False
KIE_SERVER_CONTROLLER_HOST	**KIE_SERVER_CONTROLLER_HOST**	KIE server controller host (Used to set the org.kie.server.controller system property)	my-app-controller-ocpuser.os.example.com	False
KIE_SERVER_CONTROLLER_PORT	**KIE_SERVER_CONTROLLER_PORT**	KIE server controller port (Used to set the org.kie.server.controller system property)	8080	False

Variable name	Image Environment Variable	Description	Example value	Required
DROOLS_SERVER_FILTER_CLASSES	**DROOLS_SERVER_FILTER_CLASSES**	KIE server class filtering (Sets the org.drools.server.filter.classes system property)	true	False
KIE_MBEANS	**KIE_MBEANS**	KIE server mbeans enabled/disabled (Sets the kie.mbeans and kie.scanner.mbeans system properties)	enabled	False
KIE_SERVER_HOSTNAME_HTTP	**HOSTNAME_HTTP**	Custom hostname for http service route. Leave blank for default hostname, e.g.: <application-name>-kieserver-<project>.<default-domain-suffix>	–	False
KIE_SERVER_HOSTNAME_HTTPS	**HOSTNAME_HTTPS**	Custom hostname for https service route. Leave blank for default hostname, e.g.: secure-<application-name>-kieserver-<project>.<default-domain-suffix>	–	False
KIE_SERVER_USE_SECURE_ROUTE_NAME	**KIE_SERVER_USE_SECURE_ROUTE_NAME**	If true, will use secure-APPLICATION_NAME-kieserver vs. APPLICATION_NAME-kieserver as the route name.	false	False
KIE_SERVER_HTTPS_SECRET	–	The name of the secret containing the keystore file	kieserver-app-secret	True

Variable name	Image Environment Variable	Description	Example value	Required
KIE_SERVER_HTTPS_KEYSTORE	**HTTPS_KEYSTORE**	The name of the keystore file within the secret	keystore.jks	False
KIE_SERVER_HTTPS_NAME	**HTTPS_NAME**	The name associated with the server certificate	jboss	False
KIE_SERVER_HTTPS_PASSWORD	**HTTPS_PASSWORD**	The password for the keystore and certificate	mykeystorepass	False
KIE_SERVER_BYPASS_AUTH_USER	**KIE_SERVER_BYPASS_AUTH_USER**	KIE server bypass auth user (Sets the org.kie.server.bypass.auth.user system property)	false	False
KIE_SERVER_CONTAINER_DEPLOYMENT	**KIE_SERVER_CONTAINER_DEPLOYMENT**	KIE Server Container deployment configuration in format: containerId=groupId:artifactId:version|c2=g2:a2:v2	rhdm-kieserver-hellorules=org.openshift.quickstarts:rhdm-kieserver-hellorules:1.4.0-SNAPSHOT	True
SOURCE_REPOSITORY_URL	–	Git source URI for application	https://github.com/jboss-container-images/rhdm-7-openshift-image.git	True
SOURCE_REPOSITORY_REF	–	Git branch/tag reference	master	False
CONTEXT_DIR	–	Path within Git project to build; empty for root project directory.	quickstarts/hello-rules/hellorules	False
GITHUB_WEBHOOK_SECRET	–	GitHub trigger secret	–	True

Variable name	Image Environment Variable	Description	Example value	Required
GENERIC_WEBHOOK_SECRET	–	Generic build trigger secret	–	True
MAVEN_MIRROR_URL	–	Maven mirror to use for S2I builds	–	False
MAVEN_REPO_ID	**EXTERNAL_MAVEN_REPO_ID**	The id to use for the maven repository, if set. Default is generated randomly.	my-repo-id	False
MAVEN_REPO_URL	**EXTERNAL_MAVEN_REPO_URL**	Fully qualified URL to a Maven repository.	–	False
MAVEN_REPO_USERNAME	**EXTERNAL_MAVEN_REPO_USERNAME**	Username to access the Maven repository, if required.	–	False
MAVEN_REPO_PASSWORD	**EXTERNAL_MAVEN_REPO_PASSWORD**	Password to access the Maven repository, if required.	–	False
DECISION_CENTRAL_MAVEN_SERVICE	**RHDMCENTR_MAVEN_REPO_SERVICE**	The service name for the optional decision central, where it can be reached, to allow service lookups (for maven repo usage), if required	myapp-rhdmcentr	False
DECISION_CENTRAL_MAVEN_USERNAME	**RHDMCENTR_MAVEN_REPO_USERNAME**	Username to access the Maven service hosted by Decision Central inside EAP.	mavenUser	False

Variable name	Image Environment Variable	Description	Example value	Required
DECISION_CENTRAL_MAVEN_PASSWORD	**RHDMCENTR_MAVEN_REPO_PASSWORD**	Password to access the Maven service hosted by Decision Central inside EAP.	maven1!	False
ARTIFACT_DIR	–	List of directories from which archives will be copied into the deployment folder. If unspecified, all archives in /target will be copied.	–	False
KIE_SERVER_MEMORY_LIMIT	–	KIE server Container memory limit	1Gi	False
KIE_SERVER_MGMT_DISABLED	**KIE_SERVER_MGMT_DISABLED**	Disable management api and don't allow KIE containers to be deployed/undeployed or started/stopped. Sets the property org.kie.server.mgmt.api.disabled to true and org.kie.server.startup.strategy to LocalContainersStartupStrategy.	true	True
KIE_SERVER_STARTUP_STRATEGY	**KIE_SERVER_STARTUP_STRATEGY**	When set to LocalContainersStartupStrategy, allows KIE server to start up and function with local config, even when a controller is configured and unavailable.	LocalContainersStartupStrategy	True
SSO_URL	**SSO_URL**	RH-SSO URL	https://rh-sso.example.com/auth	False

Variable name	Image Environment Variable	Description	Example value	Required
SSO_REALM	**SSO_REALM**	RH-SSO Realm name	–	False
KIE_SERVER_SSO_CLIENT	**SSO_CLIENT**	KIE Server RH-SSO Client name	–	False
KIE_SERVER_SSO_SECRET	**SSO_SECRET**	KIE Server RH-SSO Client Secret	252793ed-7118-4ca8-8dab-5622fa97d892	False
SSO_USERNAME	**SSO_USERNAME**	RH-SSO Realm Admin Username used to create the Client if it doesn't exist	–	False
SSO_PASSWORD	**SSO_PASSWORD**	RH-SSO Realm Admin Password used to create the Client	–	False
SSO_DISABLE_SSL_CERTIFICATE_VALIDATION	**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**	RH-SSO Disable SSL Certificate Validation	false	False
SSO_PRINCIPAL_ATTRIBUTE	**SSO_PRINCIPAL_ATTRIBUTE**	RH-SSO Principal Attribute to use as username.	preferred_username	False
AUTH_LDAP_URL	**AUTH_LDAP_URL**	LDAP Endpoint to connect for authentication	ldap://myldap.example.com	False
AUTH_LDAP_BIND_DN	**AUTH_LDAP_BIND_DN**	Bind DN used for authentication	uid=admin,ou=users,ou=exmample,ou=com	False
AUTH_LDAP_BIND_CREDENTIAL	**AUTH_LDAP_BIND_CREDENTIAL**	LDAP Credentials used for authentication	Password	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_JAAS_SECURITY_DOMAIN	**AUTH_LDAP_JAAS_SECURITY_DOMAIN**	The JMX ObjectName of the JaasSecurityDomain used to decrypt the password.	–	False
AUTH_LDAP_BASE_CTX_DN	**AUTH_LDAP_BASE_CTX_DN**	LDAP Base DN of the top-level context to begin the user search.	ou=users,ou=example,ou=com	False
AUTH_LDAP_BASE_FILTER	**AUTH_LDAP_BASE_FILTER**	LDAP search filter used to locate the context of the user to authenticate. The input username or userDN obtained from the login module callback is substituted into the filter anywhere a {0} expression is used. A common example for the search filter is (uid={0}).	(uid={0})	False
AUTH_LDAP_SEARCH_SCOPE	**AUTH_LDAP_SEARCH_SCOPE**	The search scope to use.	**SUBTREE_SCOPE**	False
AUTH_LDAP_SEARCH_TIME_LIMIT	**AUTH_LDAP_SEARCH_TIME_LIMIT**	The timeout in milliseconds for user or role searches.	10000	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_DISTINGUISHED_NAME_ATTRIBUTE	**AUTH_LDAP_DISTINGUISHED_NAME_ATTRIBUTE**	The name of the attribute in the user entry that contains the DN of the user. This may be necessary if the DN of the user itself contains special characters, backslash for example, that prevent correct user mapping. If the attribute does not exist, the entry's DN is used.	distinguishedName	False
AUTH_LDAP_PARSE_USERNAME	**AUTH_LDAP_PARSE_USERNAME**	A flag indicating if the DN is to be parsed for the username. If set to true, the DN is parsed for the username. If set to false the DN is not parsed for the username. This option is used together with usernameBeginString and usernameEndString.	true	False
AUTH_LDAP_USERNAME_BEGIN_STRING	**AUTH_LDAP_USERNAME_BEGIN_STRING**	Defines the String which is to be removed from the start of the DN to reveal the username. This option is used together with usernameEndString and only taken into account if parseUsername is set to true.	–	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_USERNAME_END_STRING	**AUTH_LDAP_USERNAME_END_STRING**	Defines the String which is to be removed from the end of the DN to reveal the username. This option is used together with usernameEndString and only taken into account if parseUsername is set to true.	–	False
AUTH_LDAP_ROLE_ATTRIBUTE_ID	**AUTH_LDAP_ROLE_ATTRIBUTE_ID**	Name of the attribute containing the user roles.	memberOf	False
AUTH_LDAP_ROLES_CTX_DN	**AUTH_LDAP_ROLES_CTX_DN**	The fixed DN of the context to search for user roles. This is not the DN where the actual roles are, but the DN where the objects containing the user roles are. For example, in a Microsoft Active Directory server, this is the DN where the user account is.	ou=groups,ou=example,ou=com	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_ROLE_FILTER	**AUTH_LDAP_ROLE_FILTER**	A search filter used to locate the roles associated with the authenticated user. The input username or userDN obtained from the login module callback is substituted into the filter anywhere a {0} expression is used. The authenticated userDN is substituted into the filter anywhere a {1} is used. An example search filter that matches on the input username is (member={0}). An alternative that matches on the authenticated userDN is (member={1}).	(memberOf={1})	False
AUTH_LDAP_ROLE_RECURSION	**AUTH_LDAP_ROLE_RECURSION**	The number of levels of recursion the role search will go below a matching context. Disable recursion by setting this to 0.	1	False
AUTH_LDAP_DEFAULT_ROLE	**AUTH_LDAP_DEFAULT_ROLE**	A role included for all authenticated users	guest	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_ROLE_NAME_ATTRIBUTE_ID	**AUTH_LDAP_ROLE_NAME_ATTRIBUTE_ID**	Name of the attribute within the roleCtxDN context which contains the role name. If the roleAttributeIsDN property is set to true, this property is used to find the role object's name attribute.	name	False
AUTH_LDAP_PARSE_ROLE_NAME_FROM_DN	**AUTH_LDAP_PARSE_ROLE_NAME_FROM_DN**	A flag indicating if the DN returned by a query contains the roleNameAttribute ID. If set to true, the DN is checked for the roleNameAttribute ID. If set to false, the DN is not checked for the roleNameAttribute ID. This flag can improve the performance of LDAP queries.	false	False
AUTH_LDAP_ROLE_ATTRIBUTE_IS_DN	**AUTH_LDAP_ROLE_ATTRIBUTE_IS_DN**	Whether or not the roleAttributeID contains the fully-qualified DN of a role object. If false, the role name is taken from the value of the roleNameAttribute Id attribute of the context name. Certain directory schemas, such as Microsoft Active Directory, require this attribute to be set to true.	false	False

Variable name	Image Environment Variable	Description	Example value	Required
AUTH_LDAP_REFERRAL_USER_ATTRIBUTE_ID_TO_CHECK	**AUTH_LDAP_REFERRAL_USER_ATTRIBUTE_ID_TO_CHECK**	If you are not using referrals, you can ignore this option. When using referrals, this option denotes the attribute name which contains users defined for a certain role, for example member, if the role object is inside the referral. Users are checked against the content of this attribute name. If this option is not set, the check will always fail, so role objects cannot be stored in a referral tree.	–	False
AUTH_ROLE_MAPPER_ROLES_PROPERTIES	**AUTH_ROLE_MAPPER_ROLES_PROPERTIES**	When present, the RoleMapping Login Module will be configured to use the provided file. This property defines the fully-qualified file path and name of a properties file or resource which maps roles to replacement roles. The format is original_role=role1,role2,role3	–	False
AUTH_ROLE_MAPPER_REPLACE_ROLE	**AUTH_ROLE_MAPPER_REPLACE_ROLE**	Whether to add to the current roles, or replace the current roles with the mapped ones. Replaces if set to true.	–	False

4.1.2. Objects

The CLI supports various object types. A list of these object types as well as their abbreviations can be found in the Openshift documentation.

4.1.2.1. Services

A service is an abstraction which defines a logical set of pods and a policy by which to access them. Refer to the container-engine documentation for more information.

Service	Port	Name	Description
${APPLICATION_NAME}-kieserver	8080	http	All the KIE server web server's ports.
	8443	https	
${APPLICATION_NAME}-kieserver-ping	8888	ping	The JGroups ping port for clustering.

4.1.2.2. Routes

A route is a way to expose a service by giving it an externally-reachable hostname such as **www.example.com**. A defined route and the endpoints identified by its service can be consumed by a router to provide named connectivity from external clients to your applications. Each route consists of a route name, service selector, and (optionally) security configuration. Refer to the Openshift documentation for more information.

Service	Security	Hostname
${APPLICATION_NAME}-kieserver-http	none	**${KIE_SERVER_HOSTNAME_HTTP}**
${APPLICATION_NAME}-kieserver-https	TLS passthrough	**${KIE_SERVER_HOSTNAME_HTTPS}**

4.1.2.3. Build Configurations

A **buildConfig** describes a single build definition and a set of triggers for when a new build should be created. A **buildConfig** is a REST object, which can be used in a POST to the API server to create a new instance. Refer to the Openshift documentation for more information.

S2I image	link	Build output	BuildTriggers and Settings
rhdm72-kieserver-openshift:1.1	**rhdm-7/rhdm72-kieserver-openshift**	**${APPLICATION_NAME}-kieserver:latest**	GitHub, Generic, ImageChange, ConfigChange

4.1.2.4. Deployment Configurations

A deployment in OpenShift is a replication controller based on a user defined template called a deployment configuration. Deployments are created manually or in response to triggered events. Refer to the Openshift documentation for more information.

4.1.2.4.1. Triggers

A trigger drives the creation of new deployments in response to events, both inside and outside OpenShift. Refer to the Openshift documentation for more information.

Deployment	Triggers
${APPLICATION_NAME}-kieserver	ImageChange

4.1.2.4.2. Replicas

A replication controller ensures that a specified number of pod "replicas" are running at any one time. If there are too many, the replication controller kills some pods. If there are too few, it starts more. Refer to the container-engine documentation for more information.

Deployment	Replicas
${APPLICATION_NAME}-kieserver	2

4.1.2.4.3. Pod Template

4.1.2.4.3.1. Service Accounts

Service accounts are API objects that exist within each project. They can be created or deleted like any other API object. Refer to the Openshift documentation for more information.

Deployment	Service Account
${APPLICATION_NAME}-kieserver	**${APPLICATION_NAME}-kieserver**

4.1.2.4.3.2. Image

Deployment	Image
${APPLICATION_NAME}-kieserver	**${APPLICATION_NAME}-kieserver**

4.1.2.4.3.3. Readiness Probe

${APPLICATION_NAME}-kieserver

```
/bin/bash -c curl --fail --silent -u '${KIE_ADMIN_USER}:${KIE_ADMIN_PWD}'
http://localhost:8080/services/rest/server/readycheck
```

4.1.2.4.3.4. Liveness Probe

${APPLICATION_NAME}-kieserver

```
/bin/bash -c curl --fail --silent -u '${KIE_ADMIN_USER}:${KIE_ADMIN_PWD}'
http://localhost:8080/services/rest/server/readycheck
```

4.1.2.4.3.5. Exposed Ports

Deployments	Name	Port	Protocol
${APPLICATION_NAME}-kieserver	jolokia	8778	**TCP**
	http	8080	**TCP**
	https	8443	**TCP**
	ping	8888	**TCP**

4.1.2.4.3.6. Image Environment Variables

Deployment	Variable name	Description	Example value
${APPLICATION_NAME}-kieserver	**DROOLS_SERVER_FILTER_CLASSES**	KIE server class filtering (Sets the org.drools.server.filter.classes system property)	**${DROOLS_SERVER_FILTER_CLASSES}**
	KIE_ADMIN_USER	KIE administrator username	**${KIE_ADMIN_USER}**
	KIE_ADMIN_PWD	KIE administrator password	**${KIE_ADMIN_PWD}**
	KIE_MBEANS	KIE server mbeans enabled/disabled (Sets the kie.mbeans and kie.scanner.mbeans system properties)	**${KIE_MBEANS}**
	KIE_SERVER_BYPASS_AUTH_USER	KIE server bypass auth user (Sets the org.kie.server.bypass.auth.user system property)	**${KIE_SERVER_BYPASS_AUTH_USER}**
	KIE_SERVER_CONTROLLER_USER	KIE server controller username (Sets the org.kie.server.controller.user system property)	**${KIE_SERVER_CONTROLLER_USER}**

Deployment	Variable name	Description	Example value
	KIE_SERVER_CONTROLLER_PWD	KIE server controller password (Sets the org.kie.server.controller.pwd system property)	**${KIE_SERVER_CONTROLLER_PWD}**
	KIE_SERVER_CONTROLLER_TOKEN	KIE server controller token for bearer authentication (Sets the org.kie.server.controller.token system property)	**${KIE_SERVER_CONTROLLER_TOKEN}**
	KIE_SERVER_CONTROLLER_SERVICE	The service name for the optional standalone controller. The application uses this service name to register with the controller. (If set, will be used to discover host and port)	**${KIE_SERVER_CONTROLLER_SERVICE}**
	KIE_SERVER_CONTROLLER_PROTOCOL	–	ws
	KIE_SERVER_CONTROLLER_HOST	KIE server controller host (Used to set the org.kie.server.controller system property)	**${KIE_SERVER_CONTROLLER_HOST}**
	KIE_SERVER_CONTROLLER_PORT	KIE server controller port (Used to set the org.kie.server.controller system property)	**${KIE_SERVER_CONTROLLER_PORT}**
	KIE_SERVER_ID	–	**${APPLICATION_NAME}-kieserver**
	KIE_SERVER_ROUTE_NAME	–	**${APPLICATION_NAME}-kieserver**
	KIE_SERVER_USE_SECURE_ROUTE_NAME	If true, will use secure-APPLICATION_NAME-kieserver vs. APPLICATION_NAME-kieserver as the route name.	**${KIE_SERVER_USE_SECURE_ROUTE_NAME}**

Deployment	Variable name	Description	Example value
	KIE_SERVER_USER	KIE server username (Sets the org.kie.server.user system property)	**${KIE_SERVER_USER}**
	KIE_SERVER_PWD	KIE server password, used to connect to KIE servers. Generated value can be a suggestion to use for thew s2i various (Sets the org.kie.server.pwd system property)	**${KIE_SERVER_PWD}**
	KIE_SERVER_CONTAINER_DEPLOYMENT	KIE Server Container deployment configuration in format: containerId=groupId:artifactId:version	c2=g2:a2:v2
	${KIE_SERVER_CONTAINER_DEPLOYMENT}	**MAVEN_REPOS**	–
	RHDMCENTR,EXTERNAL	**RHDMCENTR_MAVEN_REPO_SERVICE**	The service name for the optional decision central, where it can be reached, to allow service lookups (for maven repo usage), if required
	${DECISION_CENTRAL_MAVEN_SERVICE}	**RHDMCENTR_MAVEN_REPO_PATH**	–
	/maven2/	**RHDMCENTR_MAVEN_REPO_USERNAME**	Username to access the Maven service hosted by Decision Central inside EAP.
	${DECISION_CENTRAL_MAVEN_USERNAME}	**RHDMCENTR_MAVEN_REPO_PASSWORD**	Password to access the Maven service hosted by Decision Central inside EAP.
	${DECISION_CENTRAL_MAVEN_PASSWORD}	**EXTERNAL_MAVEN_REPO_ID**	The id to use for the maven repository, if set. Default is generated randomly.

Deployment	Variable name	Description	Example value
	${MAVEN_REPO_ID}	**EXTERNAL_MAVEN_REPO_URL**	Fully qualified URL to a Maven repository.
	${MAVEN_REPO_URL}	**EXTERNAL_MAVEN_REPO_USERNAME**	Username to access the Maven repository, if required.
	${MAVEN_REPO_USERNAME}	**EXTERNAL_MAVEN_REPO_PASSWORD**	Password to access the Maven repository, if required.
	${MAVEN_REPO_PASSWORD}	**HTTPS_KEYSTORE_DIR**	–
	/etc/kieserver-secret-volume	**HTTPS_KEYSTORE**	The name of the keystore file within the secret
	${KIE_SERVER_HTTPS_KEYSTORE}	**HTTPS_NAME**	The name associated with the server certificate
	${KIE_SERVER_HTTPS_NAME}	**HTTPS_PASSWORD**	The password for the keystore and certificate
	${KIE_SERVER_HTTPS_PASSWORD}	**KIE_SERVER_MGMT_DISABLED**	Disable management api and don't allow KIE containers to be deployed/undeployed or started/stopped. Sets the property org.kie.server.mgmt.api.disabled to true and org.kie.server.startup.strategy to LocalContainersStartupStrategy.
	${KIE_SERVER_MGMT_DISABLED}	**KIE_SERVER_STARTUP_STRATEGY**	When set to LocalContainersStartupStrategy, allows KIE server to start up and function with local config, even when a controller is configured and unavailable.

Deployment	Variable name	Description	Example value
	${KIE_SERVER_STARTUP_STRATEGY}	**JGROUPS_PING_PROTOCOL**	–
	openshift.DNS_PING	**OPENSHIFT_DNS_PING_SERVICE_NAME**	–
	${APPLICATION_NAME}-kieserver-ping	**OPENSHIFT_DNS_PING_SERVICE_PORT**	–
	8888	**SSO_URL**	RH-SSO URL
	${SSO_URL}	**SSO_OPENIDCONNECT_DEPLOYMENTS**	–
	ROOT.war	**SSO_REALM**	RH-SSO Realm name
	${SSO_REALM}	**SSO_SECRET**	KIE Server RH-SSO Client Secret
	${KIE_SERVER_SSO_SECRET}	**SSO_CLIENT**	KIE Server RH-SSO Client name
	${KIE_SERVER_SSO_CLIENT}	**SSO_USERNAME**	RH-SSO Realm Admin Username used to create the Client if it doesn't exist
	${SSO_USERNAME}	**SSO_PASSWORD**	RH-SSO Realm Admin Password used to create the Client
	${SSO_PASSWORD}	**SSO_DISABLE_SSL_CERTIFICATE_VALIDATION**	RH-SSO Disable SSL Certificate Validation
	${SSO_DISABLE_SSL_CERTIFICATE_VALIDATION}	**SSO_PRINCIPAL_ATTRIBUTE**	RH-SSO Principal Attribute to use as username.
	${SSO_PRINCIPAL_ATTRIBUTE}	**HOSTNAME_HTTP**	Custom hostname for http service route. Leave blank for default hostname, e.g.: <application-name>-kieserver-<project>.<default-domain-suffix>

Deployment	Variable name	Description	Example value
	${KIE_SERVER_HOSTNAME_HTTP}	**HOSTNAME_HTTPS**	Custom hostname for https service route. Leave blank for default hostname, e.g.: secure-<application-name>-kieserver-<project>.<default-domain-suffix>
	${KIE_SERVER_HOSTNAME_HTTPS}	**AUTH_LDAP_URL**	LDAP Endpoint to connect for authentication
	${AUTH_LDAP_URL}	**AUTH_LDAP_BIND_DN**	Bind DN used for authentication
	${AUTH_LDAP_BIND_DN}	**AUTH_LDAP_BIND_CREDENTIAL**	LDAP Credentials used for authentication
	${AUTH_LDAP_BIND_CREDENTIAL}	**AUTH_LDAP_JAAS_SECURITY_DOMAIN**	The JMX ObjectName of the JaasSecurityDomain used to decrypt the password.
	${AUTH_LDAP_JAAS_SECURITY_DOMAIN}	**AUTH_LDAP_BASE_CTX_DN**	LDAP Base DN of the top-level context to begin the user search.
	${AUTH_LDAP_BASE_CTX_DN}	**AUTH_LDAP_BASE_FILTER**	LDAP search filter used to locate the context of the user to authenticate. The input username or userDN obtained from the login module callback is substituted into the filter anywhere a {0} expression is used. A common example for the search filter is (uid={0}).
	${AUTH_LDAP_BASE_FILTER}	**AUTH_LDAP_SEARCH_SCOPE**	The search scope to use.
	${AUTH_LDAP_SEARCH_SCOPE}	**AUTH_LDAP_SEARCH_TIME_LIMIT**	The timeout in milliseconds for user or role searches.

Deployment	Variable name	Description	Example value
	${AUTH_LDAP_SEARCH_TIME_LIMIT}	**AUTH_LDAP_DISTINGUISHED_NAME_ATTRIBUTE**	The name of the attribute in the user entry that contains the DN of the user. This may be necessary if the DN of the user itself contains special characters, backslash for example, that prevent correct user mapping. If the attribute does not exist, the entry's DN is used.
	${AUTH_LDAP_DISTINGUISHED_NAME_ATTRIBUTE}	**AUTH_LDAP_PARSE_USERNAME**	A flag indicating if the DN is to be parsed for the username. If set to true, the DN is parsed for the username. If set to false the DN is not parsed for the username. This option is used together with usernameBeginString and usernameEndString.
	${AUTH_LDAP_PARSE_USERNAME}	**AUTH_LDAP_USERNAME_BEGIN_STRING**	Defines the String which is to be removed from the start of the DN to reveal the username. This option is used together with usernameEndString and only taken into account if parseUsername is set to true.
	${AUTH_LDAP_USERNAME_BEGIN_STRING}	**AUTH_LDAP_USERNAME_END_STRING**	Defines the String which is to be removed from the end of the DN to reveal the username. This option is used together with usernameEndString and only taken into account if parseUsername is set to true.
	${AUTH_LDAP_USERNAME_END_STRING}	**AUTH_LDAP_ROLE_ATTRIBUTE_ID**	Name of the attribute containing the user roles.

Deployment	Variable name	Description	Example value
	${AUTH_LDAP_ROLE_ATTRIBUTE_ID}	**AUTH_LDAP_ROLES_CTX_DN**	The fixed DN of the context to search for user roles. This is not the DN where the actual roles are, but the DN where the objects containing the user roles are. For example, in a Microsoft Active Directory server, this is the DN where the user account is.
	${AUTH_LDAP_ROLES_CTX_DN}	**AUTH_LDAP_ROLE_FILTER**	A search filter used to locate the roles associated with the authenticated user. The input username or userDN obtained from the login module callback is substituted into the filter anywhere a {0} expression is used. The authenticated userDN is substituted into the filter anywhere a {1} is used. An example search filter that matches on the input username is (member={0}). An alternative that matches on the authenticated userDN is (member={1}).
	${AUTH_LDAP_ROLE_FILTER}	**AUTH_LDAP_ROLE_RECURSION**	The number of levels of recursion the role search will go below a matching context. Disable recursion by setting this to 0.
	${AUTH_LDAP_ROLE_RECURSION}	**AUTH_LDAP_DEFAULT_ROLE**	A role included for all authenticated users

Deployment	Variable name	Description	Example value
	${AUTH_LDAP_DEFAULT_ROLE}	**AUTH_LDAP_ROLE_NAME_ATTRIBUTE_ID**	Name of the attribute within the roleCtxDN context which contains the role name. If the roleAttributeIsDN property is set to true, this property is used to find the role object's name attribute.
	${AUTH_LDAP_ROLE_NAME_ATTRIBUTE_ID}	**AUTH_LDAP_PARSE_ROLE_NAME_FROM_DN**	A flag indicating if the DN returned by a query contains the roleNameAttributeID. If set to true, the DN is checked for the roleNameAttributeID. If set to false, the DN is not checked for the roleNameAttributeID. This flag can improve the performance of LDAP queries.
	${AUTH_LDAP_PARSE_ROLE_NAME_FROM_DN}	**AUTH_LDAP_ROLE_ATTRIBUTE_IS_DN**	Whether or not the roleAttributeID contains the fully-qualified DN of a role object. If false, the role name is taken from the value of the roleNameAttributeId attribute of the context name. Certain directory schemas, such as Microsoft Active Directory, require this attribute to be set to true.

Deployment	Variable name	Description	Example value
	${AUTH_LDAP_ROLE_ATTRIBUTE_IS_DN}	**AUTH_LDAP_REFERRAL_USER_ATTRIBUTE_ID_TO_CHECK**	If you are not using referrals, you can ignore this option. When using referrals, this option denotes the attribute name which contains users defined for a certain role, for example member, if the role object is inside the referral. Users are checked against the content of this attribute name. If this option is not set, the check will always fail, so role objects cannot be stored in a referral tree.
	${AUTH_LDAP_REFERRAL_USER_ATTRIBUTE_ID_TO_CHECK}	**AUTH_ROLE_MAPPER_ROLES_PROPERTIES**	When present, the RoleMapping Login Module will be configured to use the provided file. This property defines the fully-qualified file path and name of a properties file or resource which maps roles to replacement roles. The format is original_role=role1,role2,role3
	${AUTH_ROLE_MAPPER_ROLES_PROPERTIES}	**AUTH_ROLE_MAPPER_REPLACE_ROLE**	Whether to add to the current roles, or replace the current roles with the mapped ones. Replaces if set to true.

4.1.2.4.3.7. Volumes

Deployment	Name	mountPath	Purpose	readOnly
${APPLICATION_NAME}-kieserver	kieserver-keystore-volume	**/etc/kieserver-secret-volume**	ssl certs	True

4.1.2.5. External Dependencies

4.1.2.5.1. Secrets

This template requires the following secrets to be installed for the application to run.

kieserver-app-secret

4.2. OPENSHIFT USAGE QUICK REFERENCE

To deploy, monitor, manage, and undeploy Red Hat Decision Manager templates on Red Hat OpenShift Container Platform, you can use the OpenShift Web console or the **oc** command.

For instructions about using the Web console, see Create and build an image using the Web console .

For detailed instructions about using the **oc** command, see CLI Reference. The following commands are likely to be required:

- To create a project, use the following command:

  ```
  $ oc new-project <project-name>
  ```

  For more information, see Creating a project using the CLI .

- To deploy a template (create an application from a template), use the following command:

  ```
  $ oc new-app -f <template-name> -p <parameter>=<value> -p <parameter>=<value> ...
  ```

  For more information, see Creating an application using the CLI .

- To view a list of the active pods in the project, use the following command:

  ```
  $ oc get pods
  ```

- To view the current status of a pod, including information whether or not the pod deployment has completed and it is now in a running state, use the following command:

  ```
  $ oc describe pod <pod-name>
  ```

  You can also use the **oc describe** command to view the current status of other objects. For more information, see Application modification operations.

- To view the logs for a pod, use the following command:

  ```
  $ oc logs <pod-name>
  ```

- To view deployment logs, look up a **DeploymentConfig** name in the template reference and run the following command:

  ```
  $ oc logs -f dc/<deployment-config-name>
  ```

  For more information, see Viewing deployment logs.

- To view build logs, look up a **BuildConfig** name in the template reference and run the command:

```
$ oc logs -f bc/<build-config-name>
```

For more information, see Accessing build logs.

- To scale a pod in the application, look up a **DeploymentConfig** name in the template reference and run the command:

```
$ oc scale dc/<deployment-config-name> --replicas=<number>
```

For more information, see Manual scaling.

- To undeploy the application, you can delete the project by using the command:

```
$ oc delete project <project-name>
```

Alternatively, you can use the **oc delete** command to remove any part of the application, such as a pod or replication controller. For details, see Application modification operations.

APPENDIX A. VERSIONING INFORMATION

Documentation last updated on Tuesday, May 28, 2019.