Installation guide for CP4BA 23.0.2 Operational Decision Manager

Contents

Preparing your chosen capabilities....1
Preparing databases and secrets for your chosen capabilities by running a script....1
Preparing to install Operational Decision Manager....9

Creating a production deployment.... 10
Generating the custom resource with the deployment script....10
Checking and completing your custom resource....12
Checking the cluster configuration.... 12
Configuring Operational Decision Manager.... 14
Validating the YAML in your custom resource file.... 15
Deploying the custom resource you created with the deployment script.... 16
Validating your CP4BA production deployment objects....18

Completing post-installation tasks....21
Operational Decision Manager.... 22
Installing Rule Designer.... 22
ROKS on IBM Cloud.... 23
Cloud Pak for Business Automation foundation.... 23
Business Automation Insights.... 23
Business Automation Navigator....24
Cloud Pak foundational services.... 25

Preparing your chosen capabilities

Each capability that you want to install must be prepared for before you apply your custom resource.

About this task

The following progress bar shows you where you are in the installation process.

Preparing a capability can involve many steps and more often than not involves setting up persistence and taking measures to secure the containers.

Important:

Take note of the capabilities and the optional components that you plan to install. Many capabilities include components from other patterns, so you must search for and use the preparing tasks under these capability headings to make sure that your installation is successful. For more information, see Capability patterns for production deployments.

Remember, Business Automation Navigator (BAN) is always installed in every deployment, so you need to create the database (ICNDB) and its secret (ibm-ban-secret). As a minimum, you need a supported database before you can apply a custom resource.

Preparing databases and secrets for your chosen capabilities by running a script

The `cp4a-prerequisites.sh` script is provided in the cert-kubernetes archive of the CASE package to help you prepare for an installation of Cloud Pak for Business Automation. The script generates property files for the selected capabilities in your deployment and must be run before your deployment is installed.

Before you begin

Before you use the `cp4a-prerequisites.sh` script to generate the property files, make sure that you review the requirements for the capabilities that you want to install together with your target database. This information is normally found in the preparing sections for each capability, where you can find the steps to manually create the databases. Consider your intended workload and the number of users that you want to access the services. For operational and performance reasons, it is important that network latency between the applications and the database server is as small as possible. For deployments that need to operate continuously with no interruptions in service, enable the databases for high availability (HA).

Tip: Run the `cp4a-prerequisites.sh` script in the "`property`" mode to create the property files for your selected capabilities and database. Then, take a note of the properties in these files so that you can match up these values with the configuration of your database services.

The `cp4a-prerequisites.sh` script uses the following utility tools and needs them to be installed on your client machine.

Note: If the script detects that any of the required tools are missing on the client, it reports the names and versions of the tools that are needed and provides a choice for you to install them.

- kubectl (the version that matches your OpenShift cluster version)

  If you prepared your client machine for an online deployment, then you have already installed kubectl.
- Java runtime environment (JRE 8.x is installed by the script if it is not found)

  - Java
  - keytool

    Make sure that you add keytool to your system PATH.
- OpenSSL version 1.1.1 or higher

About this task

Instead of going through the many documented steps to create the databases and secrets for the capabilities in your Cloud Pak for Business Automation deployment, you can use the script to generate the SQL statement files (scripts) and YAML template files for the secrets.

The `cp4a-prerequisites.sh` script has three modes.

property

The `property` mode supports the generation of property files for multiple database servers. The script uses a "**DB_SERVER_LIST**" key in the `cp4ba_db_server.property` file to list the number of instances, and creates the user property files (`cp4ba_user_profile.property`, `cp4ba_db_name_user.property`, `cp4ba_db_server.property`, and `cp4ba_LDAP.property`). Review and modify these files to match your infrastructure. Add values for the database server name, database names, database schema, LDAP server name, and LDAP attributes.

generate

The `generate` mode uses the modified property files to generate the DB SQL statement file and the YAML template for the secret.

validate

The `validate` mode checks whether the generated databases and the secrets are correct and ready to use in a CP4BA deployment.

After you downloaded the CASE package and extracted the `cert-kubernetes` archive, change the directory to the `scripts` folder under `ibm-cp-automation/inventory/cp4aOperatorSdk/files/deploy/crs/cert-kubernetes`. For more information about downloading `cert-kubernetes`, see Preparing a client to connect to the cluster.

The script can be run from this location and has the following options:

```
./cp4a-prerequisites.sh
Usage: cp4a-prerequisites.sh -m [modetype]
Options:
  -h  Display help
  -m  The valid mode types are: [property], [generate], or [validate]
      STEP 1: Run the script in [property] mode. Creates property files (DB/LDAP property file)
with default values (database name/user).
      STEP 2: Modify the DB/LDAP/user property files with your values.
      STEP 3: Run the script in [generate] mode. Generates the DB SQL statement files and YAML
templates for the secrets based on the values in the property files.
      STEP 4: Create the databases and secrets by using the modified DB SQL statement files and
YAML templates.
      STEP 5: Run the script in [validate] mode. Checks whether the databases and the secrets
are created before you install CP4BA.
```

All three modes can be run on the same client machine, but you can also run the `property` and `generate` modes on different clients. If you want to use different clients, then copy the temporary property file from the `property` mode with the output folder to the other client. Make a copy of the following files and put them into the downloaded `cert-kubernetes` folder on the other client:

```
cert-kubernetes/scripts/.tmp/.TEMPORARY.property
cert-kubernetes/cp4ba-prerequisites
```

Note: Some properties use an absolute path in their values. If you do copy the script to a different machine, make sure that the absolute paths are updated to match the location of the copied script.

The values of the following properties need to be modified after you copy the `cp4ba-prerequisites` folder to a different client.

```
********cp4ba_db_server.property*************
<DB_PREFIX_NAME>.DATABASE_SSL_CERT_FILE_FOLDER

********cp4ba_LDAP_server.property*************
LDAP_SSL_CERT_FILE_FOLDER

************cp4ba_user_profile.property******************
APP_ENGINE.SESSION_REDIS_SSL_CERT_FILE_FOLDER
```

If you ran the `cp4a-prerequisites.sh -m generate` command on the original client, you must run the command again after you modified the property files to re-create the SSL secret templates with the updated absolute paths.

Procedure

1. Make sure that you downloaded the `cert-kubernetes` repository to a Linux® based machine (CentOS Stream/RHEL/MacOS) or a client to a Linux-based machine.
2. Make sure that you are in the `scripts` folder under `cert-kubernetes`.
3. Log in to the target cluster as the `<cluster-admin>` user.

   Using the OpenShift CLI:

   ```
   oc login https://<cluster-ip>:<port> -u <cluster-admin> -p <password>
   ```

   On ROKS, if you are not already logged in:

   ```
   oc login --token=<token> --server=https://<cluster-ip>:<port>
   ```

4. Run the script in the "`property`" mode.

   ```
   ./cp4a-prerequisites.sh -m property
   ```

   Follow the prompts in the command window to enter the required information.

   a. Select the Cloud Pak for Business Automation capabilities that you want to install.

   b. Select the optional components that you want to include.

   c. Select `Yes` if you want to enable FIPS for your Cloud Pak for Business Automation deployment.

      Tip: The script asks this question only if you asked to check that FIPS is enabled on the cluster in the cluster admin script. The response is stored in the `cp4ba-fips-status` configMap.

      If you select `Yes`, the script creates the **`CP4BA.ENABLE_FIPS`** property in the `cp4ba_user_profile.property` file and records your selection.

      ```
      CP4BA.ENABLE_FIPS="true"
      ```

      If you select `No`, the value is stored as `false`.

      The property determines the value of the `shared_configuration.enable_fips` parameter in the custom resource.

   d. Choose the LDAP type that you want to use for the CP4BA deployment.

      By default, the LDAP is SSL enabled. You can disable SSL for the LDAP when you edit the LDAP property file. The script shows the following message:

      ```
      [*] You can change the property "LDAP_SSL_ENABLED" in the property file
      "cp4ba_LDAP.property" later. "LDAP_SSL_ENABLED" is "TRUE" by default.
      ```

   e. Enter your dynamic storage classes for slow, medium, fast file storage (RWX).

   f. Enter a block storage class name (RWO).

g. Select a deployment profile size from small, medium, or large [1 to 3]. The default is small (1).

h. Choose the database type that you want to use for the CP4BA deployment.

By default, the databases are SSL enabled. You can disable SSL for a database when you edit the database property file. The script shows the following message:

```
[*] You can change the property "DATABASE_SSL_ENABLE" in the property file
"cp4ba_db_server.property" later. "DATABASE_SSL_ENABLE" is "TRUE" by default.
```

Restriction: If you want to use Oracle or another custom database (a database that is not in the list) for Operational Decision Manager, then you must follow the steps to manually create the database and the secret for the `odm_configuration.externalCustomDatabase.datasourceRef` custom resource configuration parameter. For more information, see Configuring a custom external database .

i. Enter the alias names for all the database servers to be used by the CP4BA deployment. For example, `dbserver1,dbserver2` sets two database servers. The first server is named `dbserver1` and the second server is named `dbserver2`.

Note: The database server names cannot contain a dot (.) character.

j. If the script asks to enter the name for an existing project (namespace), then enter an existing project name.

k. Choose whether to restrict the egress access to external systems for your CP4BA deployment. By default the script restricts egress access. If you select `No`, egress access to external systems is unrestricted. If you select `Yes`, customize the egress access by adding network policies for known external systems post-installation. For more information, see Configuring cluster security.

Attention: If your deployment includes any capability other than Workflow Process Service Authoring and your target platform is Azure Red Hat OpenShift (ARO), then create the following network policy before you create your deployment.

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  name: ibm-content-operator-aro
  namespace: <namespace>
spec:
  podSelector:
    matchLabels:
      name: ibm-content-operator
  egress:
    - {}
  policyTypes:
    - Egress
```

The *<namespace>* is the target project for the CP4BA deployment.

When the script is finished, the messages include some actions. Read the next actions carefully and make sure that you complete them all before you go to the next step.

```
============== Created all property files for CP4BA ==============
[NEXT ACTIONS]
Enter the <Required> values in the property files under <extracted_cert-kubernetes_archive>/
cert-kubernetes/scripts/cp4ba-prerequisites/propertyfile
[*] The key name in the property file is created by the cp4a-prerequisites.sh and is NOT
EDITABLE.
[*] The value in the property file must be within double quotes.
[*] The value for User/Password in [cp4ba_db_name_user.property]
[cp4ba_user_profile.property] file should NOT include special characters: single quotation
"'"
[*] The value in [cp4ba_LDAP.property] or [cp4ba_External_LDAP.property]
[cp4ba_user_profile.property] file should NOT include special character '"'
```

The `propertyfile` directory has the following file structure:

```
├── cert
    ├── db
        └── <db_server_alias1>
```

```
        └── ...
    └── ldap
├── cp4ba_LDAP.property
├── cp4ba_db_name_user.property
├── cp4ba_db_server.property
└── cp4ba_user_profile.property
```

Note: The db directory contains one or more db server alias names that you specified.

5. Make sure that you are in the `propertyfile` folder under `cp4ba-prerequisites` and edit the property files as indicated by the `NEXT ACTIONS` messages from the script. Update the (`cp4ba_db_name_user.property`, `cp4ba_db_server.property`, `cp4ba_LDAP.property`, `cp4ba_user_profile.property`, and optionally `cp4ba_External_LDAP.property`) with the values in your environment.

   a. Edit the global section in the `cp4ba_user_profile.property` file as well as the other sections for each capability that you selected.

   The global section contains license properties and the needed storage classes. The FIPS enablement property and the egress property to restrict access to the internet are also present. The BAN section is always included, and the users and groups must be from your LDAP.

```
############################################################
## USER Property for CP4BA ##
############################################################
## Use this parameter to specify the license for the CP4A deployment and
## the possible values are: non-production and production and if not set, the license will
## be defaulted to production.  This value could be different from the other licenses in
the CR.
CP4BA.CP4BA_LICENSE="<Required>"
## On OCP, the script populates these three (3) parameters based on your input for
"production" deployment.
## The script populates the storage parameters based on your input.
## If you manually deploy without using the deployment script, then you must enter the
different storage classes for the slow, medium,
## and fast storage parameters. If you only have 1 storage class defined, then you can
use that 1 storage class for all 3 parameters.
## The sc_block_storage_classname is for Zen. Zen requires block storage (RWO) for the
metastoreDB.
CP4BA.SLOW_FILE_STORAGE_CLASSNAME="<my_file_classname>"
CP4BA.MEDIUM_FILE_STORAGE_CLASSNAME="<my_file_classname>"
CP4BA.FAST_FILE_STORAGE_CLASSNAME="<my_file_classname>"
CP4BA.BLOCK_STORAGE_CLASS_NAME="<my_block_classname>"

## Enable/disable FIPS mode for the deployment (default value is "false").
CP4BA.ENABLE_FIPS="false"

## Enable or disable egress access to external systems.
CP4BA.ENABLE_RESTRICTED_INTERNET_ACCESS="true"

############################################################
## USER Property for BAN ##
############################################################
## Provide the user name for BAN. For example: "BANAdmin"
BAN.APPLOGIN_USER="<Required>"
## Provide the user password for BAN.
BAN.APPLOGIN_PASSWORD="<Required>"
## Provide LTPA key password for BAN deployment.
BAN.LTPA_PASSWORD="<Required>"
## Provide keystore password for BAN deployment.
BAN.KEYSTORE_PASSWORD="<Required>"
## Provide the user name for jMail used by BAN. For example: "jMailAdmin"
BAN.JMAIL_USER_NAME="<Optional>"
## Provide the user password for jMail used by BAN.
BAN.JMAIL_USER_PASSWORD="<Optional>"
```

   Some values like a connection point name and table spaces for the Workflow object store initialization of Content can be any string, but are needed for the deployment to identify them.

   Important: The passwords that are used for `CONTENT.LTPA_PASSWORD` and `BAN.LTPA_PASSWORD` must be the same in the `cp4ba_user_profile.property` file.

   b. If you entered a single database server, then the DB_SERVER_NAME is set automatically. If you need more than one server, edit the *<DB_SERVER_NAME>* property prefixes in the `cp4ba_db_name_user.property` file to assign each component to a database server name

defined in the **DB_SERVER_LIST** key name in the cp4ba_db_server.property file. The value of a *<DB_SERVER_NAME>* in the username property file must match a value that is defined in the **DB_SERVER_LIST**.

c. Enter the required values for the LDAP variables in the cp4ba_LDAP.property file.

Replace the <Required> string with your existing LDAP server parameters, its query objects, users, and groups.

Important: If your target platform is ROKS Virtual Private Cloud (VPC), you can validate the connection to your LDAP only by using a VM client of the ROKS VPC. Set the LDAP server to the internal IP address or DNS of the ROKS VPC. For example, if the IP address of your LDAP is 10.240.0.16, then change the LDAP_SERVER property in the cp4ba_LDAP.property file to this address.

```
## The name of the LDAP server to connect
LDAP_SERVER="10.240.0.16"
```

If your client is not connected to the ROKS VPC, you can still set the IP address to propagate the value to the custom resource.

d. All names and passwords in the cp4ba_db_name_user.property file must be entered manually.

Replace the <yourpassword> strings with your database user passwords.

Restriction: The username values cannot contain special characters. Special characters include the equal sign (=), a forward slash (/), a colon (:), a single dot (.), single quotation marks ('), and double quotation marks ("). If a value does contain a special character, the script fails to parse the value.

The password can contain special characters, except the single quotation mark ('). The single quotation is used to enclose the string that contains special characters. If you have a password without special characters use the string as plain text. For example, if you want the password to be mypassword, specify the password as "mypassword".

```
dbserver1.GCD_DB_USER_NAME="GCDDB"
dbserver1.GCD_DB_USER_PASSWORD="mypassword"
```

If you have a password with special characters (&passw0rd), you must encode this string and add {Base64} before you add the value to the property. To encode the password on Linux, run the following command:

```
# echo -n '&passw0rd' | base64
JnBhc3N3MHJk
```

Add the encoded value to the property with the {Base64} prefix.

```
dbserver1.GCD_DB_USER_NAME="GCDDB"
dbserver1.GCD_DB_USER_PASSWORD="{Base64}JnBhc3N3MHJk"
```

6. When the user property files are complete and ready, make sure that you are in the scripts folder under cert-kubernetes, and run the cp4a-prerequisites.sh script in the "generate" mode.

```
./cp4a-prerequisites.sh -m generate
```

Note: If the script detects that the property files do not have custom values, the script stops and displays messages to help identify the missing values:

```
Change the prefix "<DB_SERVER_NAME>" in propertyfile/cp4ba_db_name_user.property to assign
which database is used by the component.
Found invalid value(s) "<Required>" in property file "propertyfile/
cp4ba_db_name_user.property", please input the correct value.
```

The following messages are displayed at the end of the execution:

```
[INFO] The DB SQL statement files for CP4BA are in directory <extracted_cert-
kubernetes_archive>/cert-kubernetes/scripts/cp4ba-prerequisites/dbscript, you can modify or
use the default settings to create the database. (PLEASE DO NOT CHANGE DBNAME/DBUSER/
```

```
DBPASSWORD DIRECTLY)
[NEXT ACTIONS]
Enter the correct values in the YAML templates for the secrets under <extracted_cert-
kubernetes_archive>/cert-kubernetes/scripts/cp4ba-prerequisites/secret_template
...
```

The /cp4ba-prerequisites directory has the following structure and varies depending on the capabilities that you selected when you ran the script:

```
├── create_secret.sh
├── dbscript
    ├── <component>
        ├── <database_type>
            ├── <db_server_alias>
                └── <sql_template>
├── propertyfile
├── secret_template
    ├── <component>
        └── <yaml_secret>
    └── ibm-ldap-bind-secret.yaml
```

7. Check that you have all the necessary files for your CP4BA deployment. Copy the required database and LDAP certificates into the target directories as indicated by the NEXT ACTIONS messages from the script. Make sure that the scripts and the YAML files have the correct values.

   For more information, see Importing the certificate of an external service.

   Note: If you selected Operational Decision Manager, the chosen database tables are automatically created when the pods are started. Therefore, the createODMDB.sql script does not include these commands.

8. When you are ready, you or a database administrator can then run the DB scripts against your database servers and use the YAML files to create the necessary secrets in your OpenShift Container Platform cluster.

   Remember: The target databases must meet the requirements of the capabilities that you want to install. Review the preparing sections for each capability before you go ahead and create the databases. For more information, see Preparing your chosen capabilities.

   a. Run the following command to go to the target CP4BA namespace.

   ```
   oc project <namespace>
   ```

   Note: If you deployed the operators in a separate namespace to the target CP4BA namespace, you must create the secrets in the CP4BA namespace.

   b. Make sure that you are in the scripts folder under cert-kubernetes and run the create_secret.sh script to apply all the YAML template files that you generated.

   ```
   ./cp4ba-prerequisites/create_secret.sh
   ```

9. Optional: When all the required databases and secrets are created, make sure that you are in the scripts folder under cert-kubernetes, and run the cp4a-prerequisites.sh script again in the "validate" mode.

   ```
   ./cp4a-prerequisites.sh -m validate
   ```

   You can also run the cp4a-prerequisites.sh script in the validate mode from inside the cp4a-operator. Complete the following steps to run the script in the validate mode from inside the cp4a-operator:

   a. Run the following command to go to the CP4BA namespace:

   ```
   oc project <namespace>
   ```

   b. Run the following command to get the CP4BA pod name:

   ```
   oc get pods | grep cp4a-operator
   ```

c. Run the following command to create an archive file of `cert-kubernetes`:

```
tar -czf <filename>.tgz cert-kubernetes
```

d. Run the following command to copy the file into the operator in the directory `/opt/ibm`:

```
oc cp <filename>.tgz <namespace>/<podname>:/opt/ibm
```

e. Run the following command to extract the archive file inside the `cp4a-operator`:

```
oc rsh <podname>
cd /opt/ibm
tar -xzf <filename>.tgz
```

f. Change the directory to the `cert-kubernetes/scripts` folder:

```
cd cert-kubernetes/scripts
```

g. Run the script in the `validate` mode:

```
./cp4a-prerequisites.sh -m validate
```

The command validates that the storage classes that you entered in the `property` mode meet the RWX and RWO requirements. If the validation is successful, it is marked as PASSED!

The command also checks that the required secrets are found and submits a validation query to the LDAP server and the list of remote database servers. If the operation succeeds within the timeout threshold, the validation is marked as PASSED! No queries are run and no data is changed, the script just reports that the connection succeeded.

If a connection is not successful, then a message informs you which connection failed. To resolve the issue, check the values in your property files so that you can correct them and try again.

Note: The `cp4a-prerequisites.sh -m validate` command uses a simple JDBC method to test the connection to the remote database servers with the Cloud Pak for Business Automation default JDBC drivers.

If you need to use customized JDBC drivers in your production deployment, you can locate these drivers by using the **`sc_drivers_url`** parameter during the configuration of the custom resource. For more information, see Optional: Preparing customized versions of JDBC drivers and ICCSAP libraries .

Results

Tip: You can change (add or remove) the selected CP4BA capabilities that you want to prepare for by rerunning the script and merging the new property files with the backed-up property files.

You can rerun the script in the "`property`" mode to create new property files. When the script detects it ran before, the previous property folder is renamed into a new time-stamped folder. The name of the backed-up folder is `cert-kubernetes/scripts/cp4ba-prerequisites-backup/propertyfile_%Y-%m-%d-%H:%M:%S`.

Use the following steps to update your property files to include your updated capabilities:

1. Copy the file `.tmp/.TEMPORARY.property` into a back up file, for example `.TEMPORARY.property.backup`.
2. Rerun the `cp4a-prerequisites.sh` script in the "`property`" mode, and choose a different selection of capabilities.
3. Restore the `cp4ba_LDAP.property` and `cp4ba_External_LDAP.property` files from the backup folder by copying and pasting them into the new folder.
4. Compare the `cp4ba_db_server.property` file from the backup folder and merge it where necessary with the new `cp4ba_db_server.property` file.
5. Merge the new `cp4ba_db_name_user.property` and `cp4ba_user_profile.property` files with the backed-up property files.

6. Rerun the `cp4a-prerequisites.sh` script in the "`generate`" mode to update the database SQL statements and YAML templates for the secrets.
7. Compare and merge the `.TEMPORARY.property.backup` file with the `.tmp/.TEMPORARY.property` file for the new capabilities.
8. Run the database SQL statements for the new capabilities.
9. Create the secrets for the new capabilities.

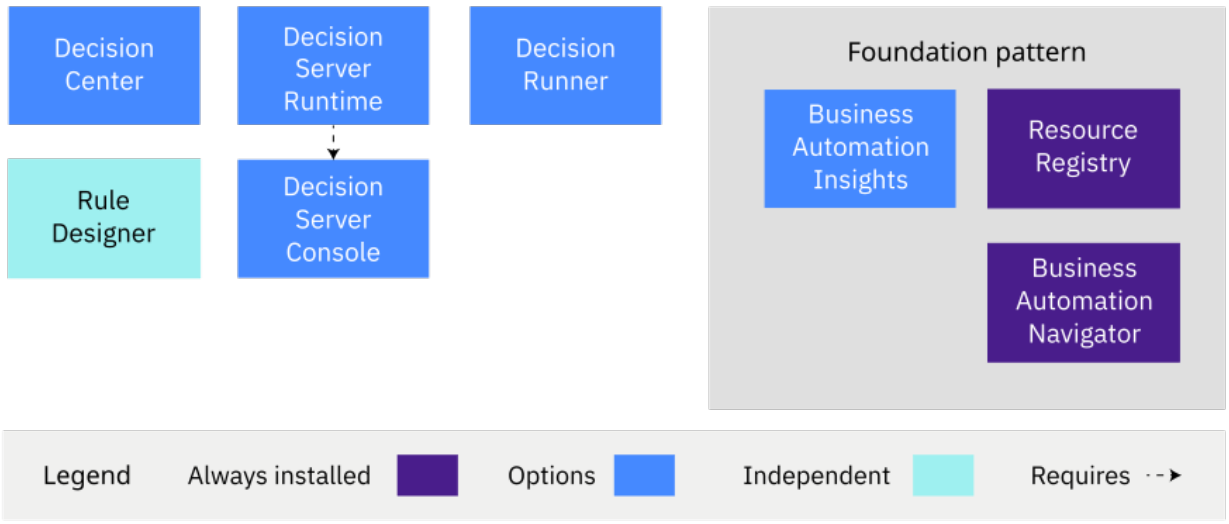
If you already installed a CP4BA deployment and want to update it with the new databases and secrets for the new capabilities, you must run the `cp4a-deployment.sh` again to update the custom resource. Do not forget to verify the custom resource YAML before you scale down the deployment, apply the new custom resource with the `--overwrite=true` parameter, and scale the deployment back up. For more information, see Stopping your deployment and Applying the upgraded custom resource .

Preparing to install Operational Decision Manager

Before you install Operational Decision Manager, check your environment to make sure that you have everything that you need. As a minimum, you must make a note of the custom resource parameter values that are needed for a data source and an LDAP configuration.

About this task

The following diagram highlights the options that you have when you select to install the `decisions` pattern. You must select one or more components, and you must then make sure that you prepare everything that is needed by these components before you apply your custom resource.



Tip: Decision Center is typically required for development and testing environments. Decision Server Runtime is typically required for testing and production environments and for using Business Automation Insights.

Choose at least Decision Center or Decision Server Runtime to have a minimum environment configuration.

You must configure a database for Operational Decision Manager to persist data. The database schema is then created automatically in the configured database.

A Lightweight Directory Access Protocol (LDAP) server and Operational Decision Manager customization can also be prepared.

If you want to create your own decision services from scratch, install Rule Designer from the Eclipse Marketplace.

Creating a production deployment

You can deploy a custom resource file from the OpenShift console, or you can create a custom resource file by running the deployment script.

About this task

The following progress bar shows you where you are in the installation process.

Generating the custom resource with the deployment script

Depending on the capabilities that you want to install, the deployment script generates a custom resource file with the selected automation containers.

About this task

The script creates a custom resource (CR) file to deploy by the Cloud Pak operator. The deployment script prompts the user to enter values to get access to the container images and to select what is installed with the deployment.

Note: The deployment script uses a custom resource (CR) template file for each pattern. The pattern template names include "`production`" and are found in the `cert-kubernetes/descriptors/patterns` folder. The CR files are configured by the deployment script.

Remember: You can run the scripts on an amd64/x86, a Linux on Z, or a Linux on Power based cluster.

Procedure

1. Log in to the cluster with the cluster administrator that you used in Preparing for a production deployment or a non-administrator user who has access to the project.

   Using the Red Hat OpenShift CLI:

   ```
   oc login https://<cluster-ip>:<port> -u <cluster-admin> -p <password>
   ```

2. View the list of projects in your cluster to see the target project before you run the deployment script.

   ```
   oc get projects
   ```

   Note: If you used the `All namespaces` option to install the Cloud Pak operator, then you must have another project in addition to `openshift-operators` in the cluster before you create the deployment. Change the scope to the project that you created for your deployment (`cp4ba-project`).

   ```
   oc project <project_name>
   ```

   The specified project is used in all subsequent operations that manipulate project-scoped content.

3. If you need to, download the `cert-kubernetes` repository to a Linux based VM/machine.

   For more information about downloading `cert-kubernetes`, see Preparing for a production deployment.

4. Run the deployment script from the local directory where you downloaded the `cert-kubernetes` repository, and follow the prompts in the command window.

Note:

Multiple capabilities can be selected and deployed in the same namespace. The script proposes new options each time you select a capability. Some capabilities are mutually exclusive, so these combinations are never proposed. For more information about the capabilities and their dependencies, see Capability patterns for production deployments.

If you select Workflow Runtime and Workstreams, two server instances are created, one server for each capability. If you want to install both capabilities into one server instance, you need to create the CR by hand and use the fully customizable (FC) template (`ibm_cp4a_cr_production_FC_workflow-workstreams.yaml`).

Warning: The script deletes any previously generated custom resource files under the `cert-kubernetes/scripts/generated-cr` folder. If you want to keep them, make copies of the YAML files somewhere else.

```
cd cert-kubernetes/scripts
./cp4a-deployment.sh
```

Remember: The `cert-kubernetes` folder is under `[PATH_TO_EXTRACTED_FILES]/ibm-cp-automation/inventory/cp4aOperatorSDK/files/deploy/crs`.

a. Accept the license. Agree to the license that is found in `cert-kubernetes/LICENSE`.

b. If you already deployed a CP4BA FileNet Content Manager instance in your namespace, then select `Yes`. The default is `No`.

c. Select the `Production` deployment type.

   The `cp4a-deployment.sh` script checks your local file system for the property files that are created by the `cp4a-prerequisites.sh` script. If the following files are found, then the script assumes that you ran the `cp4a-prerequisites.sh` script, generated the database scripts and secrets, and validated that the property values that you entered are correct for your deployment.

   ```
   cert-kubernetes/scripts/.tmp/.TEMPORARY.property
   cert-kubernetes/scripts/cp4ba-prerequisites/propertyfile/cp4ba_db_name_user.property
   cert-kubernetes/scripts/cp4ba-prerequisites/propertyfile/cp4ba_db_server.property
   cert-kubernetes/scripts/cp4ba-prerequisites/propertyfile/cp4ba_LDAP.property
   cert-kubernetes/scripts/cp4ba-prerequisites/propertyfile/cp4ba_user_profile.property
   ```

d. Select the platform type: ROKS (1) or OCP (2).

e. If you need to, provide a URL to the `.zip` file that contains your custom JDBC or IBM Content Collector for SAP Applications drivers.

   For more information, see Optional: Preparing customized versions of JDBC drivers.

f. A summary of your selection is displayed. Click "Yes" to verify that the information is correct.

Results

A custom resource file is created `scripts/generated-cr/ibm_cp4a_cr_final.yaml`.

Tip: You can rename the file and move it to another folder, or you can continue to use the file from this location. Because the maximum length of labels in Kubernetes is 63 characters, be careful with the lengths of your CR name and instance names. Some components can configure multiple instances, each instance must have a different name. The total length of the CR name and an instance name must not exceed 24 characters, otherwise some component deployments fail. When your deployment contains Workflow Authoring, the total length of the CR name cannot exceed 22 characters.

Checking and completing your custom resource

A custom resource YAML is a configuration file that describes an instance of a deployment that is created by one or more of the CP4BA operators and includes parameters to install some or all of the Cloud Pak capabilities.

About this task

A single custom resource file is used to include all of the capabilities that you want to install with a particular operator. Each time that you need to make an update or modification, you must apply the changes to your deployments. When you apply a new custom resource to an operator, you must make sure that all previously deployed resources are included, otherwise the operator deletes them.

Note: Do not modify the **appVersion** parameter when you configure the YAML file. Change the parameter only if you upgrade from a previous version.

Checking the cluster configuration

You must check and edit the shared sections of the compiled custom resource file before you apply it to the operator.

About this task

In all cases, check the <Required> values for the **image_pull_secrets** and **images** parameters in the **shared_configuration** section. For more information, see Shared configuration parameters.

Table 1. Checking <Required> parameters for selected images

Parameter	Description
dbcompatibility_init_container	Repository from where to pull the Application Engine `init_container` and the corresponding tag.
image_pull_secrets	Secrets in your target namespace to pull images from the specified repository.

Procedure

1. Locate the **shared_configuration** section in the custom resource (CR) file that you created in “Generating the custom resource with the deployment script” on page 10, then check and correct the deployment parameters.

   The custom resource templates can include the following parameters:

   License parameters

   Note: You can ignore the **sc_deployment_license** and **sc_deployment_baw_license** parameters if they do not appear in your custom resource.

   - **sc_deployment_license**, which can be `non-production`, or `production`.
   - **sc_deployment_fncm_license**, which can be: `user`, `non-production`, or `production`.
   - **sc_deployment_baw_license**, which can be: `user`, `non-production`, or `production`.

   Note: You can ignore the `fncm` and baw parameters if they do not appear in your custom resource.

   Platform parameters

   - **sc_deployment_platform**, which can be "OCP" or "ROKS".
   - **sc_ingress_enable** must be set to `true` to create an ingress on ROKS.

Sizing parameters

sc_deployment_profile_size, which determines the profile of your deployment. The default is `small`, but you can change the profile to `medium` or `large`.

Storage parameters

These parameters are mandatory.

- sc_slow_file_storage_classname
- sc_medium_file_storage_classname
- sc_fast_file_storage_classname
- sc_block_storage_classname

2. Optional: Configure the root secret, external SSL/TLS certificate secret, and the trusted certificate list.

   The custom YAML file includes the **root_ca_secret**, **external_tls_certificate_secret**, and **trusted_certificate_list** parameters. The **root_ca_secret** parameter is the name of the secret that contains the root CA signer certificate for the Cloud Pak. If the secret does not exist, then a self-signed signer certificate is generated. For more information, see Providing the root CA certificate.

   For production environments, it is likely that you want to use your own certificates that are trusted by your clients. The **external_tls_certificate_secret** parameter is used to store a wildcard certificate, which can be more convenient than a certificate for each subdomain. A multi-domain wildcard certificate can also be used to secure multiple domains and their subdomain names. For more information, see Providing certificates for external routes.

   Important: If you choose to use self-signed certificates, certain features of the product might not work as expected because of modern browser restrictions that are related to self-signed certificates. A browser blocks any redirect to a site that uses a certificate that is not signed by a root CA that is trusted by the browser. This can result in access issues for business applications.

   The **trusted_certificate_list** parameter can be used to trust root CA certificates for external services. For more information, see Connecting securely with external services.

3. Check the **resource_registry_configuration** section.

   Automatic backup for the Resource Registry is recommended. For more information, see Enabling Resource Registry disaster recovery.

4. Check the values for the **image_pull_secrets** parameter, the **sc_image_repository** parameter, and for the Application Engine repositories.

   All components use the same docker image repository. By default, the IBM Entitlement Registry is used "`cp.icr.io`".

   ```
   shared_configuration:
     sc_image_repository: cp.icr.io
     image_pull_secrets:
     - ibm-entitlement-key
   ```

   For an air gap installation, make sure that the **sc_image_repository** parameter is set to the default value. The `images` section is needed only if your environment is offline (air gapped deployment).

   ```
   shared_configuration:
     sc_image_repository: cp.icr.io
     image_pull_secrets:
     - ibm-entitlement-key
     images:
        dbcompatibility_init_container:
          repository: <registry_url>:5000/<namespace>/dba-dbcompatibility-initcontainer
          tag: <version>
        pull_policy: IfNotPresent
   ```

   The *<version>* number is 23.0.2.

   Note: The `images` section is needed only if your environment is offline (air gapped deployment).

5. **Might be required:** If you did not run the `cp4a-prerequisites.sh` script, then enter the parameter values for your data source instance in the **datasource_configuration** section.

Your deployment might need several databases. Follow the configuring instructions for each component to complete this section.

6. Optional: Modify the default value (`cpadmin`) of the **`sc_iam.default_admin_username`** parameter for the Identity Management (IM) foundational service.

   The IM admin username cannot be the same as a user in your LDAP. If your LDAP has a user with the name `cpadmin`, then set a different default admin username for IM.

7. Optional: Modify the network configuration for your deployment by adding your custom values to the **`shared_configuration.sc_egress_configuration`** section.

   By default, all the CP4A pods have restricted network access to external systems. Set the value of `sc_restricted_internet_access` to `false` to allow all CP4BA pods access to external systems. You can customize the OpenShift DNS and API Server namespaces and ports, or use the defaults. For more information, see Shared configuration parameters.

   For more information about customizing your network security, see Configuring cluster security.

Configuring Operational Decision Manager

The installation of Operational Decision Manager can be customized by changing and adding configuration parameters. The default values are appropriate to a production environment, but it is likely that you want to configure at least the security of your deployment.

About this task

ODM for production includes four containers corresponding to the following services:

- Decision Center Business console
- Decision Server console
- Decision Server Runtime
- Decision Runner

Choose which containers you want to install and which customizations you want to apply. Nonproduction deployments might include all of the four containers, whereas production deployments might have only runtime containers and a Decision Server console container.

Note: The Decision Server console cannot be installed on its own. It is automatically installed when you install the Decision Server Runtime or the Decision Runner components.

To gain insights into the decision-making process of your applications, add Business Automation Insights to your Operational Decision Manager instance to emit events from the execution of rulesets. The generated events are sent to Business Automation Insights to be processed.

Operational Decision Manager runs on Liberty, which has a `server.xml` file for the configuration. ODM authorization is managed with a `webSecurity.xml file` that provides a binding between Operational Decision Manager roles and IAM groups.

Make a note of the different customization parameter names and values. Keep these names and values close to hand for when you edit the custom resource (CR) YAML file.

Procedure

1. Open the CR file that you created in “Generating the custom resource with the deployment script” on page 10.
2. Check the values to make sure they are the values that you want to deploy.

   If you selected the `decisions` pattern, you see this value in the **`sc_deployment_patterns`** parameter.

You can also see in all cases the options to enable the ODM containers in the **odm_configuration** section.

```
# To enable ODM runtime.
    decisionServerRuntime:
      enabled: true
# To enable the authoring part.
    decisionRunner:
      enabled: true
    decisionCenter:
      enabled: true
```

3. If you need to configure more parameters, then use the fully customizable decisions template, `ibm_cp4a_cr_production_FC_decisions.yaml`, to copy lines from and paste them into your CR file.

   Go to the relevant folder in `cert-kubernetes/descriptors/patterns` to find all of the templates. For more information about downloading `cert-kubernetes`, see Preparing a client to connect to the cluster.

What to do next

Continue to configure the other capabilities that are in your CR file, and make sure that you complete the last step "Validating the YAML in your custom resource file" on page 15 before you apply the CR to the operator.

Validating the YAML in your custom resource file

You must validate your custom resource (CR) file before you apply it. It is likely that you edited the file multiple times, and possibly introduced errors or missed values during your customizations.

About this task

A good check before you apply the custom resource (CR) is to validate the YAML is executable. The web has many tools that you can use, but YAML Lint is simple and reliable.

If you want to use a command line tool, yq is a lightweight tool to help you check your YAML files. For more information, see yq - portable yaml processor.

When yq is installed, you can run the following command to verify that the custom resource has no errors.

```
yq <custom-resource-file>.yaml
```

Warning: Your CR file might contain confidential and sensitive information. Remove all of your security parameters and their values before you paste the contents of the file into the **YAML Lint** tool.

Procedure

1. Go to YAML Lint.
2. Copy and paste the contents of your CR file (without your security parameters) into the gray text box, and click **Go**.

Results

The tool reports whether the YAML is valid. If the YAML is valid, it strips out the comments and displays the configuration in a UTF-8 format. Skim the configuration again and check for any values that still show "<Required>". If you kept the comments in your CR, checking it without these lines can make it easier to read.

If the YAML is not valid, the tool generates a message and indicates the line where the error is found.

What to do next

Correct any errors that are reported, and when your CR file has no errors make sure that the CR that you intend to use includes your security parameters.

Deploying the custom resource you created with the deployment script

To install the deployment, you must apply the custom resource to the operator.

Before you begin

Make sure that you followed the instructions to prepare your environment for all of the capabilities you want to install, and you have access to all of the container images. For more information, see Getting access to images from the public IBM Entitled Registry.

Warning: If your target cluster is ROKS classic and the worker nodes rebooted, then you must synchronize the time on each of the worker nodes before you deploy the CP4BA custom resource. To synchronize the times on the worker nodes, run the following command from a connected client:

```
oc get no -l node-role.kubernetes.io/worker --no-headers -o name | xargs -I {}
--  oc debug {} -- chroot /host sh -c 'systemctl restart chronyd'
```

Procedure

1. Check that all the capabilities that you want to install are configured. If you selected CP4BA capabilities, the custom resource file is named `ibm_cp4a_cr_final.yaml`.

   ```
   cat generated-cr/<custom-resource-file>.yaml
   ```

2. Use the OpenShift CLI to deploy the configured capabilities and apply the custom resource. For CP4BA capabilities, use the custom resource file `ibm_cp4a_cr_final.yaml`.

   ```
   oc apply -f generated-cr/<custom-resource-file>.yaml
   ```

Results

The operator reconciliation loop can take some time. You must verify that the automation containers are running.

1. You can open the operator log to view the progress. Using the OpenShift CLI:

   ```
   oc logs <operator pod name> -c operator -n <project-name>
   ```

   Get the full syntax by entering the `help` command.

   ```
   oc logs --help
   ```

2. Monitor the status of your pods from the command line. Using the OpenShift CLI:

   ```
   oc get pods -w
   ```

3. When all of the pods are "Running", you can access the status of your services with the following OCP CLI command.

   ```
   oc status
   ```

See Troubleshooting to access the operator logs.

What to do next

When all of the containers are running, you can access the services.

1. Go to the `cert-kubernetes` directory on your local machine.

   ```
   cd cert-kubernetes
   ```

   For more information about downloading `cert-kubernetes`, see Preparing for a production deployment.

2. Log in to the cluster with the non-administrator user. Using the OpenShift CLI:

   ```
   oc login
   ```

3. Look for the status field of each capability by running an `oc get` command.

   ```
   oc get ICP4ACluster <instance_name> -o=jsonpath='{.status.components.<component_id>}'
   ```

   Where the *<component_id>* can be any of the following ids:

   ```
   status:
     components:
       ae-icp4adeploy-workspace-aae
       viewone
       gitgatewayService
       css
       adsMongo
       contentDesignerRepoAPI
       adsLtpaCreation
       adsCredentialsService
       workflow-authoring
       graphql
       adsRrRegistration
       adsRuntimeService
       ae-icp4adeploy-pbk
       app-engine
       contentProjectDeploymentService
       contentDesignerService
       adsGitService
       cmis
       adsParsingService
       bastudio
       ier
       adsRestApi
       adsBuildService
       navigator
       baw
       odm
       cpe
       iccsap
       tm
       adsFront
       adsRunService
       prereq
       adsRuntimeBaiRegistration
       resource-registry
       pfs
       adsDownloadService
       ca
       baml
       extshare
   ```

4. Get the access information by running either of the following commands:

   ```
   oc get cm <instance_name>-cp4ba-access-info -o=jsonpath='{.data.<component_id>-access-info}'
   ```

   ```
   oc describe icp4acluster <instance_name> -n <namespace>
   ```

   Note: The `bastudio-access-info` section provides access information for the Cloud Pak dashboard (Zen UI) and Business Automation Studio, which is installed by several patterns. The included URLs and credentials can be used to access the applications designers of the installed components.

Business Automation Studio uses the IBM Cloud Pak Platform UI (Zen UI) to provide a role-based user interface for all Cloud Pak capabilities. Capabilities are dynamically available in the UI based on the role of the user that logs in. The URL for the Admin Hub is included in the `cp4ba-access-info` ConfigMap.

You have two options to log in, **Enterprise LDAP** and **IBM provided credentials (cpadmin only)**. To log in to the Admin Hub to configure the LDAP, then click **IBM provided credentials (cpadmin only)**. You can get the details for the IBM-provided `cpadmin` user by getting the contents of the `platform-auth-idp-credentials` secret in the namespace used for the CP4BA deployment.

```
oc -n <namespace> get secret platform-auth-idp-credentials \
-o jsonpath='{.data.admin_password}' | base64 -d && echo
```

If you want to log in using the configured LDAP, then click **Enterprise LDAP** and enter the `cp4admin` user and the password in the `cp4ba-access-info` ConfigMap. The `cp4admin` user has access to Business Automation Studio features.

If you want to add more users, you need to log in with the Zen UI administrator. The *kubeadmin* user in the Red Hat OpenShift authentication and the IBM-provided `cpadmin` user have the Zen UI administrator role.

You can change the default password at any time. For more information, see Changing the cluster administrator password.

After you created a deployment, the operator automatically connects your LDAP to IAM. The users and groups you defined in your LDAP are now available via IAM.

At this point, you must associate your users and groups to Zen roles to be able to use them in all of the applications. IBM Automation has four roles that are defined: **Automation Administrator**, **Automation Analyst**, **Automation Developer**, and **Automation Operator**. For more information, see Roles and permissions.

Log in to the Common Web UI to get the IBM Cloud Pak console route and admin's password. Use the Platform UI (Zen) to create a group for your *CP4BA Developers*, and add your LDAP users and groups to this group. You then need to assign the Zen group with the **Automation Developer** role.

For more information about adding users, see Completing post-deployment tasks for Business Automation Studio.

Note: If you included multiple capabilities from FileNet Content Manager (FNCM), Automation Document Processing (ADP), and Business Automation Application (BAA) in your CP4BA deployment, then use the **Navigator for CP4BA** heading in the `cp4ba-access-info` ConfigMap and the custom resource status fields to find the route URL for Business Automation Navigator.

Validating your production deployment

Instead of manually checking all the URLs and certificates that are created by your deployment, you can run a script to validate these objects automatically in a few minutes.

Before you begin

Make sure that your client machine can connect to the cluster you want to use, and has the necessary tools. Install the appropriate tools from the following list.

jq (a JSON processor open source tool)

- On macOS:
  1. How to install jq.

     ```
     brew install jq
     ```

  2. Verify the installation.

```
jq –version
jq --help
```

- On CentOS/RHEL:
  1. Install the EPEL Repository.

     ```
     sudo yum install epel-release
     ```

  2. Update the packages.

     ```
     sudo yum update
     ```

  3. Install jq.

     ```
     sudo yum install jq
     ```

  4. Verify the installation.

     ```
     jq --version
     jq --help
     ```

About this task

The post-installation script (`cp4a-post-install.sh`) is found in the `cert-kubernetes` repository. The script helps you to assess the readiness of your CP4BA deployment, and to retrieve and validate connection information to all its services. For more information about downloading `cert-kubernetes`, see Preparing for a production deployment.

After you downloaded the CASE package and extracted the `cert-kubernetes` archive, change directory to the `/scripts` folder under `ibm-cp-automation/inventory/cp4aOperatorSdk/files/deploy/crs/cert-kubernetes`. The `/scripts` folder is the root folder ($ROOT_FOLDER) of the post-installation script.

The `cp4a-post-install.sh` script has four modes for a production deployment type:

precheck
The `precheck` mode gets some basic information about the cluster, the console, and the client.

status
The `status` mode gets the status of all the components from the custom resource (`Ready` | `Not Ready` | `Not Installed`).

console
The `console` mode gets the console connection information from the URLs and credentials.

probe
The `probe` mode checks the readiness and health of the deployment endpoints.

The script can be run with the following options:

```
./cp4a-post-install.sh --help
   --precheck         This mode gives information about the cluster and the client.
   --status           This mode gives the status of the services of the deployment.
   --console          This mode gives the service URLs of the consoles in the deployment.
   --probe            This mode checks the readiness of the deployment endpoints.
```

When you run the script in the `status`, `console`, or `probe` mode, the commands display information about the Cloud Pak for Business Automation version and interim fix number of the installed deployment. The output also includes the list of CP4BA capabilities that are installed.

If no CP4BA production deployments are found on the cluster, then information about the cluster is displayed along with the following message:

```
No resources found for CP4BA Production deployment types.
```

If no CP4BA production deployments are found in the namespace in which you run the script, then information about the cluster is displayed along with the following message:

```
No CP4BA Production deployment found in namespace NAMESPACE.
```

Procedure

1. Make sure that you downloaded the `cert-kubernetes` repository to a Linux based machine (CentOS Stream/RHEL/MacOS) or a client to a Linux-based machine.
2. Make sure that you are in the $ROOT_FOLDER under `cert-kubernetes`.
3. Log in to the target cluster as the `<cluster-admin>` user.

   Using the Red Hat OpenShift CLI:

   ```
   oc login https://<cluster-ip>:<port> -u <cluster-admin> -p <password>
   ```

   On ROKS, if you are not already logged in:

   ```
   oc login --token=<token> --server=https://<cluster-ip>:<port>
   ```

4. Check the values for the environment variables in the `/helper/post-install/env.sh` script under the $ROOT_FOLDER.

   The variable that is the most important to check is for the foundational services namespace, as it can be customized and can be either cluster-scoped or namespace-scoped. The default value is set to `ibm-common-services`, which is for a cluster-scoped instance. If you installed foundational services in a namespace-scoped instance, you must change the value for your CP4BA deployment. The following example sets `cp4ba-project` as the namespace-scoped instance.

   ```
   CP4BA_COMMON_SERVICES_NAMESPACE=cp4ba-project
   ```

5. Run the script in the "`precheck`" mode.

   ```
   ./cp4a-post-install.sh --precheck
   ```

6. Run the script in the "`status`" mode.

   ```
   ./cp4a-post-install.sh --status
   ```

   The status of the listed components can be one of the following values:

   - `Ready` is used to indicate that the component is installed successfully and ready to use.
   - `Not Ready` is used to indicate that the component is not installed yet and is not ready to be used.
   - `Not Installed` is used to indicate that the component is not included in the CP4BA deployment.
   - `Not found` is used when the status value of the resource is missing. A `Not found` value usually indicates that the operator is changing the status.

7. Run the script in the "`console`" mode.

   ```
   ./cp4a-post-install.sh --console
   ```

   The output provides valuable information about the available consoles, which can be shared and distributed to administrators and users who request access to the Cloud Pak for Business Automation capabilities.

8. Run the script in the "`probe`" mode to know whether you can access and send requests to the deployment endpoints.

   a. Add your values to the following parameters in the `env.sh` file, which is located under the $ROOT_FOLDER/`helper/post-install` folder.

PROBE_USER_API_KEY
A Platform API key that is generated for a specified user. For more information, see Generating API keys for authentication.

PROBE_USER_NAME
The name of the user in the API key. This user must have access rights to all the Cloud Pak for Business Automation applications.

PROBE_USER_PASSWORD
The basic authentication password of the user.

PROBE_VERBOSE
Can be empty or enabled with '-v' to include extra debugging information. Extra log information can be useful to determine why an endpoint is not ready to receive traffic.

b. Then, run the following command:

```
./cp4a-post-install.sh --probe
```

Note: Keep in mind that when you run the probe, OpenShift sends traffic to the pods only if the probe succeeds.

The probe mode returns one of the following messages for each endpoint:

OK
The probe succeeded. The endpoint is accessible by using a cURL command and the name of the application is returned.

BAD: <tested URL>
The probe failed. The endpoint is not accessible by using a cURL command. You can run the command from a command terminal to check the endpoint is still inaccessible.

Unauthorized: <tested URL>
The probe failed due to user authorization. You can run the script again to check the credentials are consistently rejected.

Not installed
Access to the endpoint failed. The cause of the failure can be `Not Found`, `Forbidden`, or `Service Unavailable`.

Cannot verify
The cURL command cannot be used to verify the endpoint. You must use a web browser to validate the endpoint.

What to do next

Attention: If you see connection errors to the external services, such as databases or an LDAP, after you installed your CP4BA deployment, then create an egress network policy to allow communication between your CP4BA deployment and the external services. The connection errors occur only when the external services are on the same cluster as your CP4BA deployment and in different namespaces. For more information, see Network policies to manage access to external services (egress) .

Completing post-installation tasks

For some deployments, extra steps are needed to ensure that the environment works correctly.

About this task

The following progress bar shows you where you are in the installation process.

Completing post-installation tasks for Operational Decision Manager

After you install an instance of Operational Decision Manager, review its services and install Rule Designer.

Installing Rule Designer

As a rule developer, you work in Rule Designer to create a decision service. You define a XOM and set up a vocabulary to author and orchestrate the rules that implement your business logic. You must install Rule Designer into an existing Eclipse.

About this task

Rule Designer is installed from the Eclipse Marketplace into an existing instance of Eclipse 2022-06. Use at least Eclipse Modeling Tool or Eclipse IDE for Enterprise Java Developers.

Note:

If you install Rule Designer in an Eclipse environment that does not use an English locale:

- The Eclipse messages remain in the language of your Eclipse environment.
- If Rule Designer provides the verbalizer for the language, you can create rule artifacts in this language. If a verbalizer for this language does not exist, the rule artifacts that you create are in English.
- Rule Designer messages appear in the locale if it is supported. Otherwise, they appear in English.
- To change the locale so that everything is in English, add the `osgi.nl=en_US` property to the `<Eclipse_InstallDir>/configuration/config.ini` file, and then restart Rule Designer.

Procedure

1. Start your instance of Eclipse.
2. Click **Help** > **Eclipse Marketplace**.
3. In the **Find** field, enter the text ODM and click **Go**.
4. Locate the entry of **IBM Operational Decision Manager for Developers v8.12.0 - Rule Designer** that matches the version to install, and then click **Install**.

   Eclipse calculates the dependencies and requirements.
5. In the **Install** dialog, click **Confirm**.
6. In the **Review Licenses** dialog, select **I accept the terms of the license agreement**, and then click **Finish**.

   The installation might take a couple of minutes to complete. If a security warning about the validity of the software opens, click **OK** to proceed with the installation.
7. If you are prompted to restart Eclipse, click **Yes**.
8. If not displayed by default, switch to the **Rule** perspective (click **Window** > **Open Perspective** > **Rule**).
9. Optional: If you have an existing project, import it into the Rule Designer (click **File** > **Import** > **General** > **Existing Projects into Workspace**) and copy them into the workspace.
10. To be able to publish to Decision Center from an instance of the ODM for production Helm chart, you must either import the default security certificate or use a replacement. For more information, see Importing a security certificate in Rule Designer.

Results

Using this installation of Rule Designer, you can do the following activities:

- Write a Java™ XOM that references XOM annotations and the decision engine API.
- Run and debug decision operations.
- Publish and synchronize rule projects from Rule Designer to Decision Center. To create the connection between the two in Rule Designer, add `/decisioncenter` to the Decision Center URL. For more information, see Storing and synchronizing rules.

Restriction: You cannot generate a Java client project to execute rulesets through the rule engine API or the Rule Execution Server API.

Completing extra post-installation tasks on ROKS

For deployments on Red Hat OpenShift Kubernetes Service (ROKS), routes are automatically generated. If you want to change the CA certificate, then extra steps are needed to ensure that the environment works correctly.

About this task

The access information ConfigMap includes the list of routes that can be used by external sources to access the deployed services. For information about the routes and the services they access, including URL path prefixes to use with the Zen front door, see the capability post-installation tasks.

If you enabled ingress, you can get a list of ingress objects by running the following command.

```
ibmcloud oc cluster get --cluster <clusterID> | grep Ingress
```

Completing post-installation tasks for Cloud Pak for Business Automation foundation

All capabilities contain one or more components from the Cloud Pak foundation pattern. You must check to see which components are included with your capability and complete the post-installation tasks.

Completing optional post-installation tasks for Business Automation Insights

Optionally, complete your deployment as necessary for your environment needs.

Importing sample data for IBM Business Automation Insights

As a convenience, you can test and explore IBM Business Automation Insights by using the provided script and sample data.

About this task

For prerequisites and step-by-step instructions, see https://github.com/icp4a/bai-data-samples.

Giving access to the Flink web interface through an Red Hat OpenShift route

Optional: For the Flink web user interface to be accessible from outside the cluster, set the appropriate configuration parameter in the IBM Cloud Pak® for Business Automation custom resource.

About this task

Access to the Flink user interface is helpful if no events are flowing to Elasticsearch. Flink job logs report errors, which you can explore to diagnose and resolve problems, as indicated in Troubleshooting Apache Flink jobs.

Warning: Opening access to the Flink interface might introduce a security vulnerability if no protection is set up at ingress level. Therefore, enable access only for debugging or monitoring purposes. The Flink web interface is not supported nor recommended for production.

Procedure

1. For a route to the Flink web interface to be automatically created, set the **`spec.bai_configuration.flink.create_route`** parameter to `true` in the custom resource.

```
spec:
  bai_configuration:
    flink:
      create_route: true
```

2. Retrieve the Flink web user interface URL and credentials.

   For instructions, see

   Accessing Business Automation Insights services.

Completing post-installation tasks for Navigator

If you included Navigator in your container deployment, you must do some additional configuration to ensure that the application works with your content services environment.

About this task

After you deploy your Navigator container, use the Navigator administration console to update settings for the container environment.

Procedure

1. Update the security settings on the Content Platform Engine domain so that Navigator users can access it:
   a) Log in to the Administration Console for Content Platform Engine.
   b) From the navigation pane, open the domain for editing.
   c) Click the **Security** tab.
   d) Click **Add User/Group Permissions**.
   e) Add #ALL-AUTHENTICATED to the domain, with the following settings:
      - **Source**: Direct
      - **Permission Type**: Allow
      - **Permission Group**: Custom, with **View all properties**, **Modify all properties**, **Read permissions**, and **Modify permissions** enabled.
      - **Apply to**: **This Object and Immediate Children**
2. Add the Daeja® Viewer license files to the configuration overrides directory for Navigator.
   For example, `icn-cfgstore-pv/configDropins/overrides`.

3. Update the Daeja Viewer log file path, if necessary.

   The default log file is `icn-logstore-pv/logs/daeja.log`. Confirm that the path is updated to reflect your container deployment location.

   For example: `/opt/ibm/viewerconfig/logs/daeja.log`.
4. Update the Sync Services URL:
   a) In the Navigator administration tool, click **Sync Services**.
   b) Update the default value for the public service URL.

      The URL must include the server IP address and port for the environment.

      For example: `http://IP_Address:30557/sync/notify`.
5. (Optional) Improve security for session cookies by adding httpSession configuration to your `overrides` directory.

   Create an XML file (for example, `zHTTPsession.xml`) with the following content:

```
<server>
   <httpSession
    cookieName="JSESSIONID"
    cookieSecure="true"
    cookieHttpOnly="true"
    cookiePath="/"
   >
   </httpSession>
</server>
```

   Note: Some Navigator features are affected by this setting:

   Applets
   The cookieHttpOnly=“true” setting can cause applets to fail. If you plan to use applets, remove this entry from the XML file. Or you can use the HTML-based solution, such as the HTML step processor.

   Additional Navigator features
   For information on what features are affected by this setting and possible mitigation, see the following information in the IBM Documentation for Content Navigator.
6. (Optional) If needed, change the Navigator mode.

   Navigator has two modes:

   - **Platform** mode (mode 1) displays BA Studio applications.
   - **Platform and Content** (mode 2) displays Navigator features as well as BA Studio applications.

   To change the setting, in the Navigator admin desktop, go to **Settings** > **General** tab, and use the radio options to update the **Platform Mode** setting.

Changing default values for Cloud Pak foundational services

Each instance of Cloud Pak for Business Automation that you install includes Cloud Pak foundational services, or locates an already installed instance. The foundational services can be configured post-installation to better integrate with Cloud Pak for Business Automation.

About this task

The foundational services help you manage and administer IBM software on your cluster. For example, the Cloud Pak foundational services include services such as the Platform UI (Zen) Service, the License Service, and the Identity Management (IM) Service. From the Platform UI (Zen), you can view key metrics for components of IBM Cloud Paks and Cloud Pak foundational services that are installed on the cluster.

You might want to configure the foundational services in the following ways.

Cluster administrator password

The cluster administrator password is stored in a Kubernetes secret. You can change the auto-generated password and restart the services that use the password by running a `cloudctl` command.

```
cloudctl pm update-secret kube-system platform-auth-idp-credentials -d admin_password
```

The password must follow the defined password rules. To list the password rules, run the following command:

```
cloudctl pm password-rules <namespace>
```

Platform UI (Zen) customization

The Zen Service is a reverse proxy that provides a common external URL to access Cloud Pak for Business Automation capabilities. For more information, see Platform UI. Zen uses specific roles to define who can access a particular interface. For more information about managing the roles and user permissions, see Managing users.

Note: Each Cloud Pak has its own documentation on the roles that apply to its interfaces. For some APIs and UIs, you might need extra authorization in addition to the Platform UI roles. Therefore, it is best to review the documentation of a particular API or UI before you use it.

The `ibm-zen-operator` manages the Zen Service, and the Cloud Pak for Business Automation operator does not manage the Zen settings after the initial creation. Customization of the Zen Service must be made directly in the Zen custom resource (CR).

You can review the settings and health of the Zen Service by running the following command, where *<namespace>* is usually the Cloud Pak for Business Automation namespace.

```
oc get zenservice -o yaml -n <namespace>
```

The route that is associated with Zen is called `cpd` and it uses secure TLS communications. If you need the root CA for an external truststore, see Exporting the Zen CA and common services CA . When you access the `cpd` route, login requests are redirected to the IM service and the `cp-console` route. You can customize the hostname and certificates for these routes. For more information, see Customizing the Cloud Pak Identity Management (IM) service .

All the TLS certificates for Cloud Pak foundational services are created during installation, but you can replace the certificate and the route hostname for the foundational services entry point. The entry point is the endpoint that is used to access the console from outside the cluster. For more information, see Customizing the Cloud Pak entry point .

License Service custom certificates

To avoid certificate issues and untrusted certificates when you access the UIs and REST APIs, you can configure a custom certificate for License Service communication.

1. Change the certificate name to `tls.crt`.
2. Change the name of the key to `tls.key`.
3. Run the following command to change the directory to where the certificate and the key are stored:

   ```
   cd <certificate_directory>
   ```

4. Create a secret by using the following command:

   ```
   licensingNamespace=$(oc get pods --all-namespaces | grep "ibm-licensing-service-" | awk
   {'print $1'})
   oc create secret tls ibm-licensing-certs --key tls.key --cert tls.crt -n $
   {licensingNamespace}
   ```

5. Edit the IBMLicensing custom resource to enable the `https` connection.

   ```
   apiVersion: operator.ibm.com/v1alpha1
   kind: IBMLicensing
   metadata:
     name: instance
   ```

```
spec:
  httpsEnable: true
```

To access the IBM Licensing Operator in the OpenShift console:

a. From the navigation menu, click **Operators** > **Installed Operators** > **IBM Licensing Operator**.

b. Click the **IBM License Service** tab.

c. Click the **IBMLicensings** `instance`, and then click the **YAML** tab.

6. To apply the custom certificate that you created, add the **`httpsCertsSource`** parameter:

```
apiVersion: operator.ibm.com/v1alpha1
kind: IBMLicensing
metadata:
  name: instance
spec:
  httpsEnable: true
  httpsCertsSource: ibm-licensing-certs
```

Default storage class

The storage class that you used for the ICP4ACluster instance is used for MongoDB. You can change the storage class name by adding the **`storageClass`** parameter with a storage class of your choice in the `OperandConfig` instance in the foundational services namespace.

Note: If you used a namespace-scoped foundational services instance then the namespace is defined in the common-service-maps configMap for the CP4BA deployment. If you used a cluster-scoped instance then the namespace is `ibm-common-services`.

```
- name: ibm-mongodb-operator
  spec:
    mongoDB:
      storageClass: <storage_class_name>
```

You can access the `common-service` instance by using the OpenShift console or by using the command-line interface (CLI).

- In the console, use the following steps:
  1. From the navigation menu, click **Operators** > **Installed Operators** > **Operand Deployment Lifecycle Manager** > **OperandConfig**.
  2. Click the overflow menu icon of the `common-service` instance, and click **Edit OperandConfig**.
- To use the CLI, run the following command:

```
oc edit OperandConfig common-service -n <namespace>
```