

How do I install ODM Silver topology with CP4BA 22.0.2?

By [Sia Sin Tay](#)

<https://community.ibm.com/community/user/automation/blogs/sia-sin-tay/2023/01/03/install-odm-silver-topo-cp4ba-2202>

This article is part of an article series around Operational Decision Manager (ODM) topologies in context of Cloud Pak for Business Automation (CP4BA). For more information about ODM environments and the topologies, see [CP4BA ODM topologies on OpenShift](#).

Table of Contents

<i>How do I install ODM Silver topology with CP4BA 22.0.2?</i>	1
1. Introduction	2
2. Installing ODM Silver topology	2
2.1 Procedure:.....	2
2.1.1 Authoring environment	4
2.1.2 Sandbox environment.....	5
2.1.3 Pre-Prod and Production environment	6
3. Configuring your cluster	7
3.1 Cloud Pak Platform UI (Zen) configuration	7
3.1.1 Basic Authentication	7
3.2 Managing TLS certificates	8
3.2.1 LDAP	8
3.2.2 Namespace.....	8
3.3 Configuring Rule Designer.....	8
3.4 Reaching out external services	9
4. Validating your deployment	9

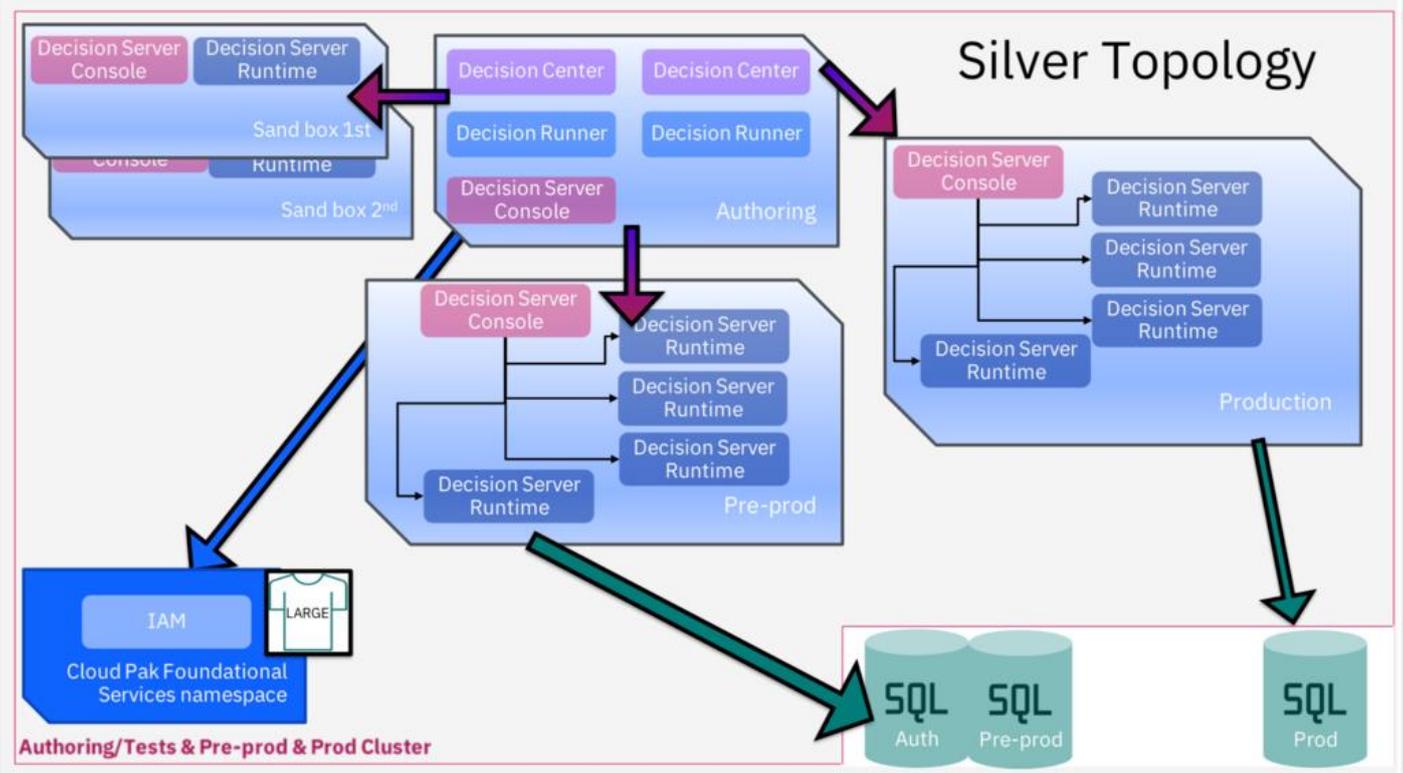
1. Introduction

This document aims at describing how to setup an ODM Silver topology deployment in the context of the Cloud Pak for Business Automation (CP4BA) 22.0.2.

The ODM Silver topology is a deployment of several ODM environments, each in an individual namespace, within a single cluster.

Minimum Silver topology consists of an Authoring, a Sand box, and a Production environment.

Full Silver topology consists of an Authoring, Sandboxes, a Pre-prod and a Production environment.



Schema of a full ODM Silver topology (fig. 1)

There is one Decision Center to govern all Decision Servers. All environments use the same Identity Access Management (IAM) for authentication and the databases are externalized.

Silver topology is best suited for applications with **medium production constraints (HA, Pre-Prod, ...)**. For more information about ODM environments and the topologies, see [CP4BA ODM topologies on OpenShift](#).

2. Installing ODM Silver topology

Silver topology is based on default Bronze topology with additional customization to allow several ODM environments within the same cluster. To install ODM Silver topology, it is recommended to start with ODM Bronze topology for 22.0.2 to setup the cluster and then obtain a baseline Cloud Pak for Business Automation deployment Custom Resource (CR) YAML file. Use this CR file and customize it per ODM environment. Other settings such as IAM configuration and certificates management are discussed in the later part of this article.

2.1 Procedure:

1. Follow the instruction in [How do I install ODM Bronze topology with CP4BA 22.0.2](#) to setup the cluster and prepare the ODM installation. Make sure to create a namespace for your ODM Silver environment. For example:

```
oc new-project <silver_topo_name>
```

2. Generate a CR file for ODM Bronze topology. For additional information, see [Option 2: Generating the custom resource with the deployment script](#).
3. Copy the CR file that is created at `scripts/generated-cr/ibm_cp4a_cr_final.yaml` and rename it as `<your_odm_env>.yaml`.

4. Assign a value to the `metadata.name` parameter in the CR file. For example, `metadata.name: authoring`
5. Set `sc_deployment_profile_size: medium` for Cloud Pak deployment profile. The deployment profile (`sc_deployment_profile_size`) of Cloud Pak for Business Automation is `small` by default. It is recommended to set to `medium` for Silver topology environments and set the IBM Cloud Platform UI (Zen) service to the same size as Cloud Pak. For more information, see [System requirements](#).
6. Remove the following unwanted parameters:
 - o `sc_deployment_fncm_license: "<Required>"`
 - o `sc_deployment_baw_license: "<Required>"`
 - o `sc_deployment_license: "<Required>"`
 - o `sc_ingress_enable: false`
 - o `sc_cpe_limited_storage: false`
 - o `sc_ingress_tls_secret_name: <Required>`
7. Fill in `image_pull_secrets` per your specific shared image pull secrets (if not so).
8. Fill in `ldap_configuration` per your LDAP configuration.
 - o  [Optional] With CP4BA 22.0.2, you can now configure multiple directories in the LDAP configuration in your CP4BA deployment and as a result authenticate users across multiple LDAPs. A mix of directory types is supported, which allows administrators to isolate teams with a specific domain. For more information, see [Support multiple LDAP directories for different user domains](#).
9. In `datasource_configuration` section, fill in `dc_odm_datasource` per your database configuration.
 - o An example for Db2 with SSL enabled:

```
datasource_configuration:
  dc_odm_datasource:
    database_servername: <db2_hostname>
    dc_common_database_instance_secret: <db2_credentials>
    dc_common_database_name: <odm_db_name>
    dc_common_database_port: '60001'
    dc_common_ssl_enabled: true
    dc_database_type: db2
    dc_ssl_secret_name: <odm-db2-ssl-cert>
    dc_ssl_enabled: true
```

- o If SSL is used to secure the database connection, set `dc_common_ssl_enabled` in the CR file to `true` and `dc_ssl_secret_name` parameter with a secret containing the Db2 SSL certificate.
- o The secret containing the Db2 server certificate is created if you have run the "`cert-kubernetes/scripts/cp4a-prerequisites.sh`" script in [Preparing databases and secrets for your chosen capabilities by running a script](#). Otherwise you can create the secret by running the following command:

```
oc create secret generic odm-db2-ssl-secret --from-file=db2-server-certificate=<your_path>/server.crt
```

- o whereby `server.crt` is the Db2 SSL certificate public key in ASCII format. For example:

```
-----BEGIN CERTIFICATE-----
MIIHdzCCBfegAwIBAgIQCKZtYygfn9pg13D0uAX YzANBgkqhkiG9w0BAQsFADBg ...
3R7IrdK8aS1WUG1KulqEDiV4TJ 1XpcoUq8wtmBSw1fyV7g=
-----END CERTIFICATE-----
```

- o For more information on how to generate the Db2 SSL certificate, see [Self-signing digital certificates](#).
- o  [Optional] If you have difference databases for your Decision Center and Decision Server instances, rather than configuring `dc_odm_datasource` parameter, you can now configure `dc_odm_decisioncenter_datasource` and `dc_odm_decisionserver_datasource`

under the `datasource_configuration` tag of the custom resource file. For more information, see [Targeting separate external databases](#).

10. Remove the parameter `deployment_profile_size: "small"`.
11. Make sure that you have a fully defined CR file for an ODM Bronze topology. You can refer to the sample as provided at the end of [How do I install ODM Bronze topology with CP4BA 22.0.2](#).
12. Modify `spec.odm_configuration` and `spec.shared_configuration.sc_optional_components` sections in the CR file according to your desired ODM environment and save the changes. See the examples below for each ODM environment.
13.  [Optional] You can configure multi-zone support by setting the `nodeAffinity` parameter. Using this parameter, the ODM service pods can be deployed on nodes in a specific zone. This way, you can organize and optimize access to the underlying resources, like storage and database, by region. For more information, see [Configuring multi-zone support](#).
14. Apply the fully defined custom CR file to install your ODM environment.

```
oc apply -f <your_odm_env>.yaml
```

15. Verify the deployment that the ODM decision pods are all ready after a couple of reconcile loops of the CP4BA operator.

2.1.1 Authoring environment

Authoring environment consists of Decision Server Console, 2 Decision Center and 2 Decision Runner. Edit your Authoring custom CR file install these components. For example:

```
metadata:
  name: authoring
...
spec:
  odm_configuration:
    decisionCenter:
      enabled: true
      replicaCount: 2
      resources:
        limits:
          cpu: '2'
          memory: 8Gi
        requests:
          cpu: '1'
          memory: 4Gi
    decisionServerRuntime:
      enabled: false
    decisionRunner:
      enabled: true
      replicaCount: 2
      resources:
        limits:
          cpu: '2'
          memory: 2Gi
        requests:
          cpu: 500m
          memory: 2Gi
    decisionServerConsole:
      resources:
        limits:
          cpu: '2'
          memory: 1Gi
        requests:
          cpu: 500m
          memory: 512Mi
...
  shared_configuration:
    sc_optional_components: decisionCenter,decisionRunner
    sc_deployment_license: production
```

2.1.2 Sandbox environment

Sandbox environment only consists of Decision Server Console and a Decision Server Runtime.

Edit your sandbox custom CR file to only install Decision Server Console and Decision Server Runtime. For this environment, assign `spec.shared_configuration_sc_deployment_license` to `non-production` as it is not a production environment.

For example:

```
metadata:
  name: sandbox
...
spec:
  odm_configuration:
    decisionCenter:
      enabled: false
    decisionServerRuntime:
      enabled: true
      replicaCount: 1
      resources:
        limits:
          cpu: '2'
          memory: 2Gi
        requests:
          cpu: '2'
          memory: 2Gi
    decisionRunner:
      enabled: false
    decisionServerConsole:
      resources:
        limits:
          cpu: '2'
          memory: 1Gi
        requests:
          cpu: 500m
          memory: 512Mi
...
  shared_configuration:
    sc_optional_components: decisionServerRuntime
    sc_deployment_license: non-production
```

2.1.3 Pre-Prod and Production environment

Production environment consists of Decision Server Console and 3 Decision Server Runtime. Pre-prod environment is similar to Production environment with a Decision Server Console and several Decision Server Runtime.

For Production environment, assign `spec.shared_configuration_sc_deployment_license` to `production`.

As for Pre-prod environment, assign `spec.shared_configuration_sc_deployment_license` to `non-production`.

Edit your Pre-prod custom CR file to install these components. For example:

```
metadata:
  name: preprod
...
spec:
  odm_configuration:
    decisionCenter:
      enabled: false
    decisionServerRuntime:
      enabled: true
      replicaCount: 3
      resources:
        limits:
          cpu: '2'
          memory: 2Gi
        requests:
          cpu: '2'
          memory: 2Gi
    decisionRunner:
      enabled: false
    decisionServerConsole:
      resources:
        limits:
          cpu: '2'
          memory: 1Gi
        requests:
          cpu: 500m
          memory: 512Mi
...
  shared_configuration:
    sc_optional_components: decisionServerRuntime
    sc_deployment_license: non-production
    # in case of production environment, set sc_deployment_license to production
    # sc_deployment_license: production
```

See [Performance Check List of OCP for CP4BA](#) to perform some additional tuning to this Production or Pre-Production environment.

3. Configuring your cluster

3.1 Cloud Pak Platform UI (Zen) configuration

User access to ODM is now managed through the Zen UI. Predefined ODM permissions and ODM roles are available. You can manage the permissions of your users for ODM in IBM Cloud Pak Platform UI (Zen). These permissions are used to access Decision Center and Decision Server console, or to control access to Decision Server Runtime and management REST API endpoints.

A Zen API key is used to allow automatic authentication of the ODM services with the Cloud Pak Zen platform. For Decision Server Runtime REST API calls, although you can use a Zen API key, it is preferable to use basic authentication for performance reasons.

For more information about Zen and the relevant configuration for ODM, refer to [Configuring user access](#).

3.1.1 Basic Authentication

By default, a basic registry with the following users is provided in the form of a `webSecurity.xml` file:

- `resExecutor` to execute rules on the Decision Server Runtime
- `odmAdmin` to execute REST API calls on Decision Center and Decision Server Console

Follow the following steps to have a customized registry for your ODM environment.

1. Copy the default registry `webSecurity.xml` file.
2. Adapt the group mappings inside the `webSecurity.xml` file depending on your requirement. Here is an example of a custom `webSecurity.xml` file for a production environment to be included in the secret `my-auth-secret`:

```
<server>
  <basicRegistry id="basic" realm="basic">
    <user name="odmAdmin" password="odmAdmin"/>
    <user name="resExecutor" password="resExecutor"/>
    <user name="resAdmin1" password="resAdmin1"/>
    <group name="resExecutors">
      <member name="resExecutor" />
      <member name="odmAdmin" />
    </group>
    <group name="basicResAdministrators">
      <member name="resAdmin1" />
    </group>
  </basicRegistry>
  <variable name="odm.resAdministrators.group1"
value="group:basic/basicResAdministrators"/>
  <variable name="odm.resMonitors.group1" value="group:basic/basicResAdministrators"/>
  <variable name="odm.resExecutors.group1" value="group:basic/resExecutors"/>
</server>
```

3. Create a secret in your project namespace using the command:

```
oc create secret generic my-auth-secret --from-
file=webSecurity.xml=<your_path>/webSecurity.xml
```

4. Pass the secret to ODM configuration through the `spec.odm_configuration.customization.authSecretRef` parameter of the CR file:

```
odm_configuration:
  customization:
    authSecretRef: my-auth-secret
```

For more information, refer to [Optional user access configurations](#).

3.2 Managing TLS certificates

Along with the LDAP connection issue that you can meet in secured connection context, the secured connection to other environments (different namespaces) is not configured by default and needs to manually push the certificate.

3.2.1 LDAP

If you use a SSL-enabled LDAP in your environment, you must create the SSL secret with the certificate of the LDAP server. Put the LDAP server certificate in the operator trust list as described in [Importing the certificate of an external service](#). For related information about securing LDAP by SSL, see [Configuring LDAP over SSL](#).

3.2.2 Namespace

By default, a secured connection between Decision Center in Authoring environment to Decision Server Console in another environment, leads to an error like the following one:

```
javax.net.ssl.SSLHandshakeException: java.security.cert.CertificateException: Path does not chain with any of the trust anchors
```

To overcome this issue, you need to import the certificate of Sandbox/Prep-prod/Production environment to Authoring environment. Here are the steps:

1. Extract the RES certificate by downloading it from the RES console (for example Production environment) using a browser.
2. Create a new secret in **Authoring** namespace (key=tls.crt with cert content):

```
oc create secret generic my-prod-env-secret --from-file=tls.crt=<your_path>/cpd.pem -n <your_authoring_namespace>
```

3. In the ICP4ACluster deployment YAML configuration of your Authoring environment, specify this secret as a trusted certificate in the `spec.shared_configuration.trusted_certificate_list` parameter. Save the changes.

```
shared_configuration:
  trusted_certificate_list:
    - my-prod-env-secret
```

4. Wait for a couple of reconcile loops of the CP4BA operator and that the ODM pods are restarted.
 - **Tips:** You can scale down CP4BA operator and scale it back up to fasten the reconciliation.
5. Check in the log of new DC pods that the production environment certificate is added to the keystore:

```
Importing trusted certificates
...
[Storing /config/security/truststore.jks]
Importing trusted certificates ./my-prod-env-secret/tls.crt
Certificate was added to keystore
```

3.3 Configuring Rule Designer

To be able to securely connect your Rule Designer to the Decision Server and Decision Center components that are running in an OCP cluster, you need to establish a Transport Layer Security (TLS) connection through a security certificate. For more information, see [Importing a security certificate in Rule Designer](#).

3.4 Reaching out external services

Last but not least, to integrate with an external service in general, you must first import its TLS certificate into the operator trust list. These certificates are added to the truststore of each component in the Cloud Pak. The procedure is described in [Importing the certificate of an external service](#).

4. Validating your deployment

To ensure that the environment is well deployed at CP4BA level, follow the steps in [Validating your production deployment](#). Additional validations can be done at ODM level using [How do I validate my ODM topology with CP4BA 22.0.2](#). You can then perform post installation tasks as described in [Completing post-installation tasks for Operational Decision Manager](#).

[#OperationalDecisionManager\(ODM\)](#) [#topology](#) [#businessrules](#) [#CloudPakforBusinessAutomation](#)