Setup and Operations Guide for SAP Digital Manufacturing for edge computing

Generated on: 2025-12-14 10:29:11 GMT+0000

SAP Digital Manufacturing | latest

Public

Original content: https://help.sap.com/docs/SAP_DIGITAL_MANUFACTURING/5a297c4405af474f91080d3969bbc26b?locale=en-US&state=PRODUCTION&version=latest

Warning

This document has been generated from SAP Help Portal and is an incomplete version of the official SAP product documentation. The information included in custom documentation may not reflect the arrangement of topics in SAP Help Portal, and may be missing important aspects and/or correlations to other topics. For this reason, it is not for production use.

For more information, please visit https://help.sap.com/docs/disclaimer.

Basic Concepts

SAP Digital Manufacturing for edge computing supports business continuity for mission-critical operations on the shop floor.

Purpose of SAP Digital Manufacturing for edge computing

SAP Digital Manufacturing for edge computing provides a buffering capability for SAP Digital Manufacturing to operate regardless of cloud outage, planned or unplanned, which may last a few hours.

It constantly synchronizes data that is critical for production (for example, orders, SFCs, recipes) from cloud to edge to prepare the edge for operations. Edge transactions are constantly synchronized to the cloud for replay. In case of an outage, transactions that are posted against edge will be synchronized to cloud when cloud is up and running.

Key Features

- Cache master data at edge (see Data Load)
- Synchronize transactions to cloud

  Note

  All system integrations, such as with EWM, are managed in the cloud when transactions are replayed.

- Execute transactions for offline scenarios
- Monitor edge transactions on cloud
- Support inbound calls by external systems, for example, Production Connector

Note

See Basic Functions for further information.

Related Information

- For more information on how to use applications provided by SAP Digital Manufacturing for edge computing, see Application Help for Edge.
- For more information on edge APIs, see APIs for SAP Digital Manufacturing for edge computing.
- For more information on edge integration scenarios, see SAP Digital Manufacturing for edge computing Integration and SAP Digital Manufacturing for edge computing - Developer's Guide.
- For more information on the security aspect of SAP Digital Manufacturing for edge computing, see Security Guide for SAP Digital Manufacturing and the Network and Communication Security for Edge under it.
- Refer to this link for notices on the free and open-source software used in SAP Digital Manufacturing for edge computing.
- For more information on the SAP Digital Manufacturing for edge computing limitations, see related section in 3379404.

Hardware and Software Requirement

You must meet the following hardware and software requirements for using SAP Digital Manufacturing for edge computing.

- Have an edge device meeting the following requirements:
  - [Minimum] 64 GB memory
  - [Minimum] 16 CPU cores
  - [Minimum] 50 GB persistent volumes
  - The operating system architecture must be Linux/amd64.

  Note

  The sizing of the system is highly dependent upon the functionalities in use and the data volumes. It's recommended to plan for suitable upgrades of the resources based on usage.

- The cluster has at least two storage classes, one with "ReadWriteOnce" (RWO) access mode and another with "ReadWriteMany" (RWX) access mode. The cluster should be able to provision persistent volumes automatically.

  Note

  Different access modes are required or supported by different services. While all services require the storage class to support the RWO access mode, the `edge-object-store` and Production Connector service requires its storage class to support the RWX mode for productive usage. Therefore, it is necessary to specify a different storage class for the `edge-object-store` or the Production Connector service than for the other services. For details, see Helm Values.

- Have a Kubernetes cluster with a supported Kubernetes version

  Note

  See Release Process and Strategy for tested versions.

Data Load

SAP Digital Manufacturing for edge computing caches data critical for production from cloud to edge.

Master data, such as BOM, routing, and recipes, along with some business data like orders, are never created, updated, or deleted on the edge. However, this data is required for various transactions such as execution, data collection, assembly/component consumption. Therefore, SAP Digital Manufacturing for edge computing requires syncing this data from the cloud to the edge to support production. Once you have installed the edge device and deployed SAP Digital Manufacturing for edge computing, you need to trigger an initial load to synchronize suitable data created or updated on the cloud to the edge. Meanwhile, after the initial load, data created or modified is constantly synchronized to the edge as part of delta load.

To trigger an initial load, you must create an initial load enablement request through a ticketing-based approach, as detailed in the sections below.

Initial Load

The initial load synchronizes existing objects from the cloud to an edge device for associated plants. You can trigger the initial load after deploying SAP Digital Manufacturing for edge computing. However, before triggering the initial load, you must submit an initial load enablement request. For details, see Enable and Trigger an Initial Load. This request requires preparing data in the object store for the initial load to the edge.

In some cases, you need to trigger the initial load again if:

- there are newly supported object types
- data in some existing object types requires an update
- some objects have failed to synchronize with the edge in the previous initial load

If you don't do that, you can only use features that rely on existing object types.

Delta Load

The system performs delta load each time you create, edit or delete an object on cloud.

> **Note**
> If you want the system to perform a data load for object types not yet supported by the initial load, you can edit the objects and save them again. This will trigger a delta load.

Supported Object Types

Choose **View Objects** on the detail page of the **Manage Edge Devices** app to view all object types related to the initial load of your selected edge device. For details on changes of the APIs related to the object types in each release, see 3586729.

> **Note**
> There are some limitations on data and relevant use cases. For details, see related section in 3379404.

Modular Deployment

In addition to the core functionalities that are the fundamental services required for the operation of SAP Digital Manufacturing for edge computing, SAP Digital Manufacturing for edge computing also provides modules—a suite of microservices and a range of backing services that can be customized through modular deployment.

A module is a feature or a package of features that can be deployed and used independently on edge. Options to un-deploy a module are also available if needed. Modules are organized under module groups based on their functionalities. In most cases, you can select the module you'd like to run on your edge device. In contrast, in other special cases, due to technical dependencies, some modules under certain groups need to be selected together to function properly. The following modules are available:

Module Group	Module Name	Edge Tag	Description
Automation	MQTT	`mqtt`	MQTT broker functionality. For details, see SAP MQTT Broker Deployment: An Introduction and SAP MQTT Broker. After choosing and configuring this module in installation, you can create a subscription with the **Manage Automatic Triggers** app to an MQTT message. When an MQTT client publishes the message to a message broker on edge and your predefined conditions are met, an action is triggered. For details, see Manage Subscriptions. **Note**

Module Group	Module Name	Edge Tag	Description
			This module is to be deprecated along with the sunset of SAP MQTT Broker in release 2602. You are recommended to use third-party MQTT brokers instead.
Automation	Production Process Orchestration	`process-orchestration`	Workflow engine that orchestrates production activities. Select this module to enable edge runtime environment production process and edge event business rule monitoring. For details, see Available Services and Subprocesses, Create and Define a Production Process, Monitor Production Processes, Monitor Event Business Rules and Edge UI Applications.
Automation	Production Connectivity Model	`production-connectivity`	Establish connectivity with Production Connector. For details, see Set Up Connectivity to Shop Floor with Production Connectivity Model and Production Connector on Edge. **Note** Outbound communication with Production Connector on Windows is not supported yet.
Automation	Production Connector	`production-connector`	Embedded middleware that communicates with shop floor systems. **Note** Select the **Production Process Orchestration**, **Production Connectivity Model**, **Production Connector** module to enable Production Connector on edge runtime environment production process. For details, see Available Services and Subprocesses, Create and Define a Production Process, Monitor Production Processes and Edge UI Applications.
Printing	Printing	`printing`	Edge printing functionality. For details on the preparation work of using the edge printing functionality, see Automatic Printing at the Edge.
UI	POD	`pod`	Production operator dashboard. This module should be enabled to access the edge landing page. See Edge UI Applications for details.

You can pick and choose the modules needed to be run on your edge devices. For details, see Create Edge Device and Helm Values.

Go to the **Manage Edge Devices** app for an overview of all module statuses on the edge device. The modules you select to install on edge should be:

1. Supported in your edge version, otherwise the installation will still be successful but the modules will not be deployed
2. Consistent with the tags value that you set to true in your values file, otherwise the installation will fail

Release Process and Strategy

Release Process

In general, SAP Digital Manufacturing for edge computing (edge) follows the same Release Schedule and Dates as SAP Digital Manufacturing (cloud).

Preview Version Release (for example, 1.2402.0-beta, 1.2402.1-beta)

For the quality testing window, following the Quality Upgrade (cloud) for an upcoming new release:

- A new Helm chart is published to the Repository-Based Shipment Channel (RBSC), with a "-beta" suffix in the version number. You need to add the `--devel` option in your Helm command to pull the Helm chart.

Patches may be delivered during this window for identified issues.

Major Release (for example, 1.2402.0)

After the quality testing window, following the Production Upgrade (cloud) for the new release:

- A new Helm chart is published to the Repository-Based Shipment Channel (RBSC).

Hotfix Release (for example, 1.2402.1)

After the new release, patches will be delivered as hotfixes, following the same hotfix process of the cloud.

> **Note**
> For some issues identified for the edge solution, hotfixes may be delivered for the cloud components, which do not require an upgrade of the Helm chart.

Cloud–Edge Compatibility

SAP supports deploying and running edge components on the latest major version and its patches. Security and functional hotfixes of edge components are only delivered for the latest available major version.

While SAP provides support for deploying and running edge components one major version behind the cloud version, it is recommended to upgrade edge components to the latest major version as soon as possible. If you need to apply a security or functional hotfix, you must upgrade edge to the latest major version:

- before Quality Upgrade (cloud) for edge instances connected to your Quality tenant
- before Production Upgrade (cloud) for edge instances connected to your Production tenant

Note that deploying or running edge components with two or more major versions behind the latest available major version is not supported.

For example, if the cloud is on version 2411, the edge must be on either version 2411 or 2408. If you upgrade the cloud to version 2502, the edge must be on at least version 2411.

Tested Kubernetes Distributions and Versions

The following Kubernetes distribution and versions were tested in the most recent release:

Functional test:

- Azure virtual environments (validation on Azure subscription):
  - 1.30.12 and higher (Ubuntu, Linux)
  - 1.31.10 and higher (Ubuntu, Linux)
- Azure Stack HCI (Arc-enabled, now called Azure Local):
  - 1.29.4 (Ubuntu, Linux)

Lifecycle management test:

- 1.31.9 (Ubuntu, Linux)
- 1.32.5 (Ubuntu, Linux)
- 1.32.6 (Ubuntu, Linux)
- 1.33.1 (Ubuntu, Linux)
- 1.33.2 (Ubuntu, Linux)

In general, newer Kubernetes versions will be supported for newer releases of edge, while older versions will no longer be supported. You are highly recommended to keep your Kubernetes versions up to date, following the Kubernetes Version Skew Policy . For details of AKS release, see AKS Kubernetes release calendar .

Install, Upgrade, and Uninstall

SAP Digital Manufacturing for edge computing is delivered as a Helm application. You can use Helm CLI directly to deploy it or use your own self-managed deployment tool, such as Argo CD, which supports Helm applications.

You can download the software from the Repository-Based Shipment Channel (RBSC) beforehand or download it while executing Helm commands.

→ Tip

As the technical user password used to pull Docker images from RBSC expires every six months, a local container registry is recommended.

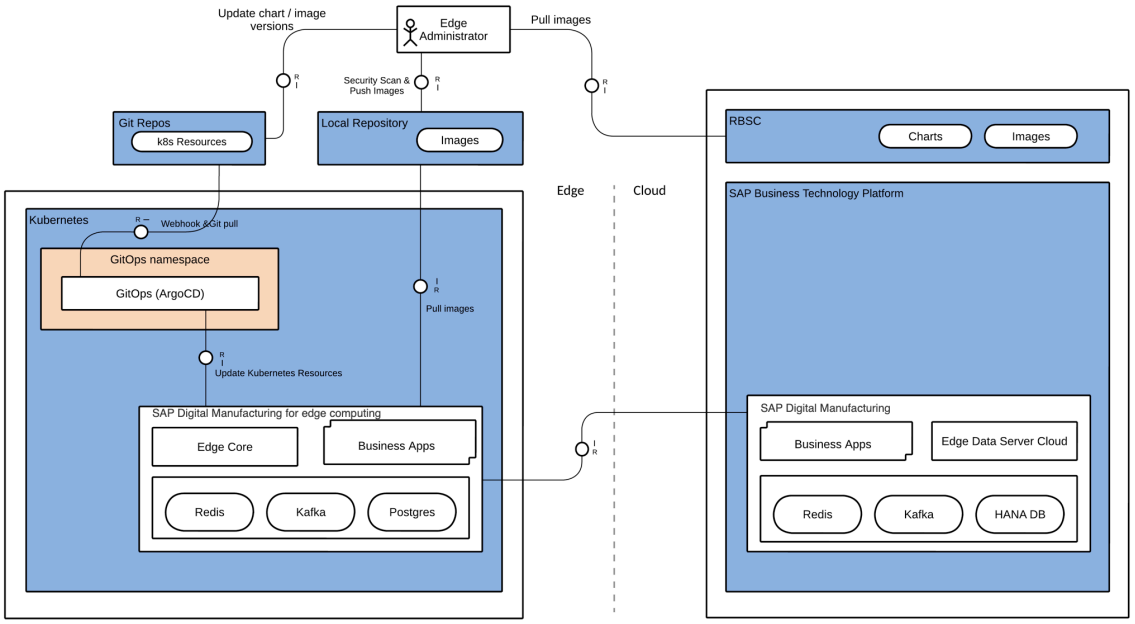
i Note

For details on how to manage installation of the SAP Digital Manufacturing for edge computing Helm chart, with a focus on the aspects that relate to the SAP MQTT Broker, see Deployment of SAP Digital Manufacturing for Edge Computing.

Architecture and Workflow Overview

The architecture of SAP Digital Manufacturing for edge computing involves publishing Helm charts and images to SAP's container registry. Edge administrators can pull and scan images before installation. A connection is established between edge and cloud, with data synchronization and reconciliation between the two.

Architecture

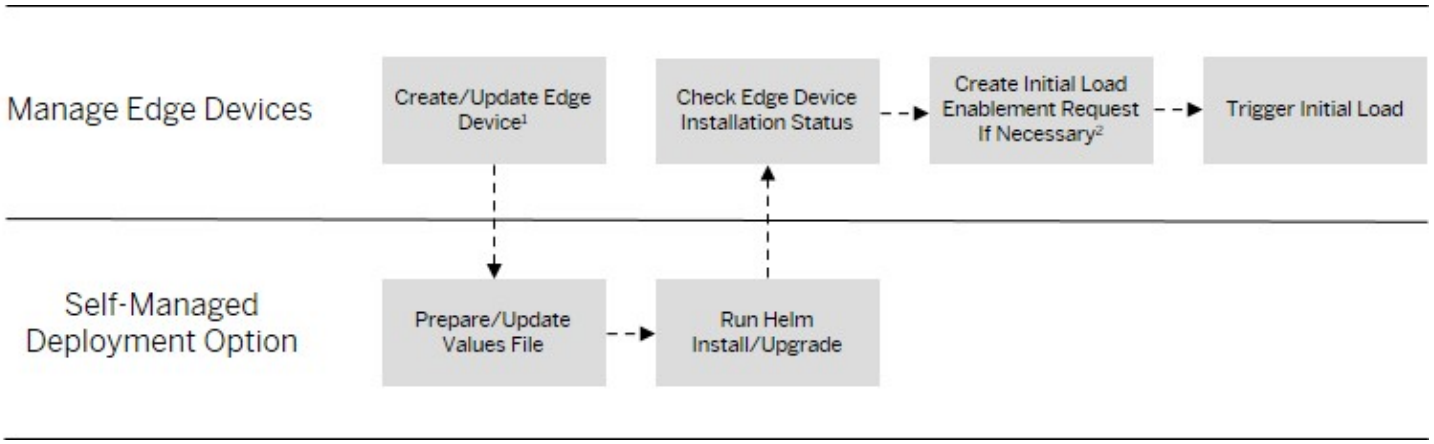


SAP Digital Manufacturing for edge computing Helm charts and images are published to SAP's public container registry—Repository-Based Shipment Channel (RBSC). As an edge administrator, you can pull the images in advance to do a security scan and push the images to your local container registry.

Before the actual installation of SAP Digital Manufacturing for edge computing, create an edge device in the **Manage Edge Devices** app to obtain additional information.

As part of the SAP Digital Manufacturing for edge computing installation process, a connection is established between edge and cloud. The edge core components and Edge Data Server in the cloud are responsible for upstream and downstream data synchronization as well as some basic functionality of edge. All edge transactions are eventually synchronized to the cloud for cloud business apps to reconcile.

Workflow



Note

1. Update your edge devices if there are module changes.
2. New data objects may be added when there's a new module deployment or a major version upgrade. Create an initial load enablement request for new data objects. Check the **Manage Edge Devices** app to see if an initial load is required.

Prerequisites

1. You have an S-user account that has download permissions.
2. Your organization's SAP customer account has a valid license for SAP Digital Manufacturing for edge computing.
3. You have acquired and set up an edge device.

   See Hardware and Software Requirement for requirements on edge devices.

Preparations

1. Configuration in your SAP Business Technology Platform (BTP) subaccount:

   a. Create a service key for an SAP Digital Manufacturing service instance and download the service key. See Prepare for API Integration.

   > **Note**
   >
   > Ensure that you remove "\n" in the verification key.

   b. If you plan to deploy the printing module, create a service key for an SAP Print Service instance and download the file.

   c. Create a role collection. See Define a Role Collection.

   d. Add the role of **Edge_Admin** to the role collection. See Add Roles to a Role Collection.

2. Configuration in the Repository-Based Shipment Channel (RBSC):

   a. Log on as an S-user.

   b. On the **Licenses** tab, copy the repository endpoint for Helm of the product DIGITAL MANUFACTURING EDGE 4.0, `https://7355500010090000683 3.helmsrv.base.repositories.cloud.sap/`.

   c. On the **Users Management** tab, create a technical user and copy its basic authentication password. This technical user will be used to pull Helm charts and download the docker images later. For more information, see Manage Technical Users in SAP Repositories Management.

   > **Caution**
   >
   > The technical user password expires every six months.

3. Ensure that the following domains are reachable from the Kubernetes cluster:

URL	Purpose
`*.authentication.{region}.hana.ondemand.com`	Communicate with the SAP Business Technology Platform XSUAA service
`{region}-prod.apps.dmc.cloud.sap`	Synchronize data between edge and cloud, for productive landscape For example, `eu20-prod.apps.dmc.cloud.sap`

URL	Purpose
`{region}-quality.apps.dmc.cloud.sap`	Synchronize data between edge and cloud, for quality landscape For example, `eu20-quality.apps.dmc.cloud.sap`
`*.repositories.cloud.sap`	Access the Repository-Based Shipment Channel (RBSC) **Note** This is required only if you pull images directly from RBSC, but not required if you use your own container registry.

Note

For the first three URLs, refer to Supported Data Centers and Languages to confirm the region. You can also find the region information in your service key or launchpad URL.

4. Ensure the cluster has an ingress that is either NGINX Ingress Controller or Istio.

Note

If you install NGINX by yourself, be sure to install the NGINX Ingress Controller published by Kubernetes. For more information, see Appendix 2: Install NGINX Ingress Controller.

See Appendix 1: Install Istio Service Mesh for details on Istio installation with Istio client `istioctl`. Istio-based service mesh add-on for Azure Kubernetes Service (AKS) is not supported.

For additional ingress controller setup in the case of MQTT, see Ingress Controller Setup.

5. The ingress should have an external IP address assigned, which is accessible from all workstations. To assign an external IP to the load balancer, ensure the cluster has a fixed IP address, an IP range, or available external IPs for dynamic allocation before installation. Additionally, map the hostname of SAP Digital Manufacturing for edge computing and `edge-ui.<hostname>` to the external IP address in the DNS server.

6. Prepare certificates.

    a. Generate or obtain a server certificate for SAP Digital Manufacturing for edge computing. Ensure that the private key is included. You'll also need the CA certificates.

    Note

    Refer to Appendix 3: Generate Self-Signed Certificates Using OpenSSL for a sample server certificate signing request configuration file.

    b. Prepare a client certificate that is signed for a technical user "dm-edge-services" as the common name. Ensure that the private key is included. You'll also need the CA certificates.

    c. Obtain the signing CA for the Cloud Connector system certificate.

    The certificates must be in PEM format, base64-encoded. The private key of the edge server certificate must be decrypted, otherwise you'll encounter the "failed to parse private key" error when later configuring Kubernetes secrets as part of the installation.

7. If you intend to use Helm to install the application, install Helm and Kubernetes command tool `kubectl` on your workstation.

8. If you use a local container registry, it is best practice to list the docker images before pulling them. This is especially important if you want to deploy a module, which is delivered as a subchart. See Get Docker Image List for details.

Database

Two types of databases are supported in SAP Digital Manufacturing for edge computing: embedded databases and external databases. You can choose to use embedded PostgreSQL or your own external database (PostgreSQL or Microsoft SQL Server) as the database when installing SAP Digital Manufacturing for edge computing. External PostgreSQL is the default option. When working with external databases, you need to use a dedicated one specifically for SAP Digital Manufacturing for edge computing. The database user must have all privileges including the schema creation privilege granted on the given database.

→ Recommendation

We strongly recommend that you use your own external database for SAP Digital Manufacturing for edge computing. We do not provide advanced functionality like high availability, backup and restore, and so on for the embedded PostgreSQL database. Use it only for proof of concept or test purposes.

You need to provide relevant information if you want to use an external database. For details, see Helm Values. Ensure that the Kubernetes cluster where you are going to install SAP Digital Manufacturing for edge computing can connect to the database.

After edge installation, switching between different database types is not supported. You need to re-install the edge if you want to do so.

Tested Database Versions

The following database versions are tested and supported:

- PostgreSQL version 14
    - Bitnami 14.5.0
- Microsoft SQL Server 2022
    - Azure SQL Database. For details, see Logical Servers .
    - Azure Arc-enabled SQL Managed Instance. For details, see SQL Server enabled by Azure Arc .

Create Edge Device

You need to first create an edge device on the cloud to obtain mandatory information like device ID and then proceed with the edge installation. To create an edge device and define its name, description, plant, and select the modules you want to use on your edge device, do the following:

Prerequisites

You have been assigned the role of `Edge_Admin`.

Procedure

1. Go to the **Manage Edge Devices** app and choose **Create**.
2. In the **General Information** tab of the **New Edge Device** page, fill in relevant fields.

- **Name**: Mandatory. The name can only contain A–Z, a–z, 0–9, hyphens, and underscores, with a length no longer than 50 characters. It must be unique per tenant, and start and end with either a letter or a number.
- **Description**: no longer than 255 characters
- **Plant**: Mandatory. You need to select at least one plant or more from all plants which have no edge device associated.

> **Note**
>
> A plant can only be assigned to one edge device. You cannot change the plant assignment after installation.

3. In the **Modules** tab, select the module you'd like to deploy.

   See Modular Deployment for details on the module.

4. Choose **Create**.
5. Choose **View Parameter Values** to see the following information in YAML format:
   - deviceId: auto-generated GUID
   - tags: module information
   - global.tenantId: your SAP Business Technology Platform tenant ID
   - connectionString: used for establishing the edge-to-cloud connection
6. Choose **Copy** to copy the values in YAML format. Alternatively, you can also choose **Save as File** to save the values into a YAML file.

   You can also see in the header section or **General Information** tab the following information:

   - Status:
     - **Awaiting Installation**: You have created the edge device but have not started installation or attempted installation but failed
     - **Installed**: Successfully installed

     > **Note**
     >
     > The "Installed" status doesn't indicate the system's health. Instead, check the system's health status using Kubernetes CLI or other tools.

   - Version: Helm chart version of SAP Digital Manufacturing for edge computing
   - Hostname
   - IP Address
   - Landing Page: URL to log into the edge landing page. For details, see Edge UI Applications.
   - Last Updated On

> **Note**
>
> If you want to edit the plant field, do it before installation.

Install with Helm

This chapter provides instructions on using Helm commands to deploy SAP Digital Manufacturing for edge computing.

Context

Helm hooks allow you to define actions that should be executed at specific points in the Helm release lifecycle. Helm hooks implemented by SAP Digital Manufacturing for edge computing are:

- pre-install
- post-install
- pre-upgrade
- post-upgrade
- pre-delete

More hooks may be implemented in the future. You need to make sure your deployment tool supports these Helm hooks.

The following procedure illustrates how to install the application using Helm.

Procedure

1. Set the Kubernetes context:

   ```
   export KUBECONFIG=[kubeconfig_file_path]
   ```

   ### ℹ Note

   For Windows, the command is `set KUBECONFIG=[kubeconfig_file_path]`.

2. Create a namespace for SAP Digital Manufacturing for edge computing:

   ```
   kubectl create namespace [namespace]
   ```

3. [Required for Istio] Enable Istio for the namespace:

   ```
   kubectl label namespace [namespace] istio-injection=enabled
   ```

4. Add a chart repository for the SAP Digital Manufacturing for edge computing Helm chart:

   a. Add a chart repository, using the repository endpoint, and the technical username and password for the RBSC or your own container registry:

   ```
   helm repo add [helm_repo] [helm_chart_URL] --username *** --password *** --pass-credent
   ```

   The `helm_repo` parameter is the name you give to a local Helm chart repository. The `helm_chart_URL` parameter is obtained from RBSC, as part of Preparations or for your own container registry.

   b. Check the latest version of the Helm chart:

   ```
   helm search repo [helm_repo]/offline-edge --versions
   ```

   ### ℹ Note

   If you are testing beta versions, which are reserved for the quality testing window before official release, and for beta features, add the "--devel" option.

   c. Check the configurable values of the chart:

```
helm show values [helm_repo]/offline-edge --version [version]
```

5. Create Kubernetes secrets to store different certificates.

Secret	Secret Content	Namespace	Helm Value	Note
Secret 1	CA certificate for SAP Digital Manufacturing for edge computing client certificate	- Istio (for Istio) - SAP Digital Manufacturing for edge computing (for NGINX)	`global.ingressCACertificate`	
	CA certificate for Cloud Connector system certificate			You need to have obtained a Cloud Connector system certificate. Self-signed system certificate is not supported.
Secret 2	SAP Digital Manufacturing for edge computing server certificate		`global.ingressServerCertificate`	
Secret 3	CA certificate for Dex server certificate	SAP Digital Manufacturing for edge computing	`global.dexCACertificate`	If you use external Dex. For details, see Configure SAP Digital Manufacturing for edge computing. If you use embedded Dex, the Dex server shares the same server certificate as SAP Digital Manufacturing for edge computing itself. For details, see Configure SAP Digital Manufacturing for edge computing.

For NGINX:

```
kubectl create secret generic [secret_1] --from-file=ca.crt=[trusted_ca_certificates] -n [dm_
kubectl create secret tls [secret_2] --key [server_certificate_key] --cert=[server_certificat
kubectl create secret generic [secret_3] --from-file=ca.crt=[Dex_CA_Certificate] -n [dm_edge_
```

For Istio:

```
kubectl create secret generic [secret_1] --from-file=ca.crt=[trusted_ca_certificates] -n [ist
kubectl create secret tls [secret_2] --key=[server_certificate_key] --cert=[server_certificat
kubectl create secret generic [secret_3] --from-file=ca.crt=[Dex_CA_Certificate] -n [dm_edge_
```

ℹ Note

For Istio, if you want to use cert-manager, secret 1 must be named as `<secret_2_name>-cacert`. For example, if secret 2 is named `tls-secret`, secret 1 must be named `tls-secret-cacert`.

6. Prepare a values file (for example, `values.yaml`) accordingly. For more information, see Helm Values.
7. Install the application using Helm:

```
helm install [release_name] [helm_repo]/offline-edge -f [values_file] -n [namespace] --versio
```

The `release_name` parameter is a name you give to the SAP Digital Manufacturing for edge computing instance to be installed.

8. After installation is complete, check the status of the following pods in the given namespace:

```
kubectl get pods -n [namespace]
```

The pods should be in **Running** status and jobs in **Completed** status.

Next Steps

Trigger an initial load to synchronize data from the cloud to the edge in the **Initial Load** tab of the **Manage Edge Devices** app. For details, see Data Load and Enable and Trigger an Initial Load.

Use User-Defined Password for Embedded Database

When installing SAP Digital Manufacturing for edge computing with an embedded database, a password is needed for the PostgreSQL database. By default, the password is randomly generated.

ℹ Note

If you switch from using a randomly generated password to using a user-defined password for an existing deployment, you can only use the existing database password.

Get the existing database password:

```
kubectl get secret postgressecret -n offline-edge -o jsonpath='{.data.DB_PASSWORD}' | base64 —d
```

However, you also have the option to set your own password.

ℹ Note

You are highly recommended to set a custom password for Argo CD deployments, as each "sync" generates a random database password, which disrupts existing deployments.

To set your own password, follow either of the options below:

- Specify the password directly using the `global.database.auth.password` parameter.
- Use an external secret. See Use External Secrets for Credentials for details.

ℹ Note

The above options also apply to external databases.

Related Information

Helm Values

Use External Secrets for Credentials

You can use an external secret for credentials when installing SAP Digital Manufacturing for edge computing.

Context

Supported credentials are:

- UAA information
- Database password
- Dex information
- Docker registry credential

Note

The external secret is applicable for both embedded and external databases.

Procedure

1. Configure a secret for these values in your external secret management system. External secret management systems may store the secrets as key–value pairs or in one single JSON string, depending on the provider.
    - **CLIENT_ID**: UAA client ID. You can obtain it from the service key of an SAP Digital Manufacturing service instance in your BTP subaccount.
    - **CLIENT_SECRET**: UAA client secret. You can obtain it from the service key of an SAP Digital Manufacturing service instance in your BTP subaccount.
    - **URL**: UAA URL. You can obtain it from the service key of an SAP Digital Manufacturing service instance in your BTP subaccount.
    - **VERIFICATION_KEY**: UAA verification key. You can obtain it from the service key of an SAP Digital Manufacturing service instance in your BTP subaccount.
    - **DB_PASSWORD**: Your password for the PostgreSQL database.
    - **OPENID_CLIENT_ID** and **OPENID_CLIENT_SECRET**: Client ID and secret configured in your Dex server for a static client.
    - **CONNECTOR_CLIENT_ID** and **CONNECTOR_CLIENT_SECRET**: ID and secret of OpenID Connect client that you configure in the IdP to represent the embedded Dex.
    - **DOCKER_REGISTRY_USERNAME**: User name for the RBSC docker registry or your local container registry.
    - **DOCKER_REGISTRY_PASSWORD**: Password for the RBSC docker registry or your local container registry.

    The following paragraphs take the external secret management systems Azure Key Vault and HashiCorp Vault as examples, including those paths and namespace.

[Azure Key Vault] Create a vault secret "offline-edge" (object type: Secrets) with a JSON string under secret/dmedge/quality.

Sample Code

```
{
"CLIENT_ID": "sample!b1472|dmc-services-quality!b330",
"CLIENT_SECRET": "Sample",
"URL": https://sample-dmcazqualityhc1.authentication.eu20.hana.ondemand.com,
"VERIFICATION_KEY": "my-dm-uaa-verification-key",
"DB_PASSWORD": "Sample_MyPssw0rd",
"OPENID_CLIENT_ID": "my-idp-oidc-client-id",
"OPENID_CLIENT_SECRET": "my-idp-oidc-client-secret",
"CONNECTOR_CLIENT_ID": "my-dex-client-id",
"CONNECTOR_CLIENT_SECRET": "my-dex-client-secret",
"DOCKER_REGISTRY_USERNAME": "Sample_UserName",
"DOCKER_REGISTRY_PASSWORD": "Sample_MyPssw0rd"
}
```

[HashiCorp Vault] Create a vault secret "offline-edge" under the namespace “cubbyhole”.

Sample Code

```
$set VAULT_ADDR=https://vault.tools.sap/
$set VAULT_TOKEN={your vault access token}

vault kv put cubbyhole/offline-edge CLIENT_ID="sample!b1472|dmc-services-quality!b330" CLIE

# check if created
$vault kv get cubbyhole/offline-edge
```

2. Synchronize secrets from the secret management system into Kubernetes with a Kubernetes operator. In this step, you create a secret store in your Kubernetes cluster, which will be integrated with the external secret management system. Refer to the official documentation for details if you use External Secrets Operator (ESO).
3. Prepare Helm values for external secrets. In the values.yaml file:

Sample Code

```
global:
  security:
    externalSecrets:
      enabled: true
      secretStoreRefKind: SecretStore
      secretStoreRefName: vault-secret-store     # Kubernetes secret store
      remoteRefKey: dm-edge  # secret maintained in the external secret management system,
      database:
        enabled: true  # if also enable external secret for db password
        remoteRefKey: key1
      dockerRegistry:
        enabled: true # if also enable external secret for docker registry credential
        remoteRefKey: key2
      serviceKey:
        enabled: true # if also enable external secret for uaa information
        remoteRefKey: key3
```

```
dex:
  enabled: true # if also enable external secret for dex information
  remoteRefKey: key4
```

Related Information

Helm Values

Enable and Trigger an Initial Load

Learn how to request initial load enablement and trigger an initial load with the **Manage Edge Devices** app. You need to perform an initial load not only when you install SAP Digital Manufacturing for edge computing for the first time, but also for module changes and new release version upgrades.

Create an Initial Load Enablement Request

Check in the **Manage Edge Devices** app if an initial load is required. To request initial load enablement, create an incident with SAP under component `MFG-DM-EDGE`. The system guides you through this process.

You need to prepare the following information:

- The system

  Provide the SAP Digital Manufacturing system URLs of the tenants you want to include.

- The plant

  Provide the plants you configure to run on the edge.

- The timestamp

  The initial load synchronizes all existing data from the cloud to your edge devices by default, regardless of the time the data is created or modified. But you also have the option to synchronize only the data modified after a specific date and time (UTC).

  All data objects, except for the Unit of Measure, support synchronization based on the date and time. You can specify different date and times on various objects.

Trigger an Initial Load

Once an initial load enablement request is processed, you can trigger the initial load and monitor its status on the detail page of the **Manage Edge Devices** app.

You can choose **View Objects** to view the details of all object types related to the initial load. Each object type can have the following statuses:

- **New**: The objects in the object type are ready to be synchronized in the initial load for the first time. Or, there are newly supported object types after the previous initial load.
- **Completed**: All objects in the object type have been successfully synchronized to the edge device in the initial load.
- **Update Required**: Data in some existing object types requires an update after the previous initial load.
- **Failed**: Some objects in the object type have failed to be synchronized with the edge in the previous initial load.

To trigger an initial data load, choose **Run** in the **Initial Load** tab of the edge device detail page.

1. You will be prompted with the object detail for this initial load.
2. Choose **Configure** in the message strip to specify a date range for order-related objects to synchronize and filter out irrelevant data if needed. In the **Configure Order-Related Objects** window, enter a number between 1 and 90, and choose **Save**.

   Order-related objects include order, recipe/routing, bill of material, and resolved routing. As order-related objects are relevant only when the order is still active, you can filter out irrelevant data to expedite the initial load. Note that the time is displayed according to your browser's date time, ignoring hours.

   ### Example

   As an admin in Dublin, you're responsible for managing an edge device for a plant in Frankfurt. When you specify to synchronize orders from the last 14 days at 2025/02/28 05:07:00 Dublin time, it means the orders in Frankfurt start from 2025/02/14 00:00:00 Dublin time. The 5-hour and 7-minute time difference is ignored for that plant.

3. Choose **Run** after confirming to run the initial load.

The initial load can have the following statuses displayed along with the percentage in the progress bar:

- **Preparing**: The data is being prepared in the cloud. This can happen concurrently with data synchronization.
- **Synchronizing**: The system is synchronizing data from the cloud to the edge device.
- **Completed**: All objects of all associated plants have successfully synchronized to the edge device.
- **Failed**: At least one object was unable to synchronize with the edge, even if other objects had been completed.

Choose □ *(Refresh)* to refresh the initial load status.

The start time is the trigger time of the initial load and the end time is the finish time of the last object, whether it is completed successfully or failed.

Note

While an initial load runs, you can't trigger new initial loads. If you uninstall the edge device during an initial load, the process terminates.

Related Information

Data Load

Enable UI Access

You can use Dex to authenticate your user with SAP Digital Manufacturing for edge computing for UI access. Users must belong to authorized user groups to access SAP Digital Manufacturing for edge computing.

Dex integrates with your own identity provider (IdP) where your users are stored and managed. You can use embedded Dex or your own Dex as the identity service. External Dex is the default option.

Related Information

Use External Dex

Use Embedded Dex

Use External Dex

Different from Dex server provided by SAP Digital Manufacturing for edge computing right out of the box, external Dex refers to a Dex server that you install yourself.

Related Information

Prerequisites
Configure Dex as Client Application in IdP
Configure User Groups in IdP
Install and Configure Dex
Configure SAP Digital Manufacturing for edge computing

Prerequisites

1. If you install Dex on a separate cluster, you must ensure that the issuer URL can be reached from the cluster where SAP Digital Manufacturing for edge computing is installed.
2. The OIDC issuer URL of your IdP must contain "https://". Dex does not support legacy or insecure issuer URLs.
3. You have mapped the domain for your Dex server to an IP address in DNS.

Configure Dex as Client Application in IdP

As Dex needs to request identity from an upstream IdP, you need to configure the following information about the Dex server in the IdP:

- Redirect or callback URI: this should match what you configure in Dex

  The IdP redirects the user back to Dex, with the callback URL carrying an authorization code or token to complete the authentication process.

- Connector type: OIDC
- Client authentication method: for example, client ID and secret for authentication, or certificate for authentication
- Ensure that your IdP sends all required OpenID Connect claims in addition to these additional claims:
  - email (string)
  - email_verified (boolean)

    Note

    Some providers return claims without "email_verified" if they don't use email verification during the enrollment process or act as a proxy for another IdP. You can override this with the following option in the Dex configuration: `insecureSkipEmailVerified`: `true`. For details, see Install and Configure Dex.

  - groups (list of strings)

  You may use the OpenID Connect Discovery endpoint to verify the claims supported by your IdP: `<issuer-url>/.well-known/openid-configuration`

If the supported claims do not contain the required claims, you need to work with your IdP administrator to configure custom claims or claim mappings.

- Grant type: Authorization code, Refresh

  > **Note**
  >
  > PKCE is not supported by Dex.

- Post logout redirect URL: this should be in the pattern of `https://edge-ui.<dm-edge-hostname>`. Refer to specific SAP Digital Manufacturing for edge computing values (`global.idpLogoutUrl`) in Helm Values.
- Refresh token and access token

  We recommend setting the refresh token validity of your IdP or Dex ID token to under 24 hours. The shorter time from the two values is used as the session time for each login of the edge UI. The maximum session time is 24 hours. However, you need to stay active during the session. If you're inactive for over 15 minutes, you'll be logged out.

  We suggest setting a shorter access token validity, like one hour.

Refer to the documentation of your IdP on how to configure service providers.

Configure User Groups in IdP

SAP Digital Manufacturing for edge computing supports two user groups with different scopes, targeted at supervisors and operators respectively. For details on permissions each authorization policy offers to apps on the edge landing page, see Authorization Policy.

You need to map these authorization policies directly to user groups defined in the IdP.

In your IdP:

1. Configure one or two user groups for your users to access UI applications. For details, see Create User Groups.
2. Assign users to the appropriate user group. For details, see Import Users for SAP Digital Manufacturing.

When installing SAP Digital Manufacturing for edge computing, define your user groups in the `global.security.amsAuthFramework.supervisorGroupName` and `global.security.amsAuthFramework.operatorGroupName` values. The values should match the user group names you've defined in the IdP.

> **→ Tip**
>
> You are recommended to configure different groups for different edge devices.

Install and Configure Dex

Configure Secrets for Dex Certificates

Communication between Dex and SAP Digital Manufacturing for edge computing is secured through certificates. In the namespace of Dex, you need to create Kubernetes secrets to store its server certificate and key. Later you will need to configure a secret for the signing CA certificate in the SAP Digital Manufacturing for edge computing namespace to establish trust. The

following step for Kubernetes secret creation is targeted at NGINX only. For instructions about other Ingress controllers, please refer to specific documentation.

1. Create a secret for the server certificate and its private key:

```
kubectl create secret tls dex-cert --key dex.key --cert dex.crt -n dex
```

Prepare Values File for Installation

Before installing Dex, prepare a values file. The following example is a sample values.yaml file for when Dex integrates with the IdP based on OIDC, using client ID and secret authentication. Replace the values in curly brackets with your own values according to the comments.

Sample Code

```
config:
  # Token issuer URI of Dex
  issuer: {dex-server-url}
  # Skip additional approval to share data with application from IdP
  oauth2:
    skipApprovalScreen: true
  storage:
    type: memory
  # Configuration information for integration with an upstream IdP
  # Below configuration assumes that Dex is integrated with the IdP based on OIDC protocols
  connectors:
  - config:
      # ID configured in the IdP for Dex as an OIDC client
      clientID: ***
      clientSecret: ***
      # Token issuer URI of the IdP
      issuer: https://{idp-domain}
      # Callback URL for Dex
      redirectURI: {dex-server-url}/callback
      # User attributes requested from the IdP during the OIDC authentication flow
      scopes:
      - profile
      - email
      userIDKey: sub # should match IdP and the Manage User Assignment app
      # Subject name attribute for users
      userNameKey: email
      # Ensures that group information is included in the ID token for DM Edge access
      insecureEnableGroups: true
      # Some providers return claims without "email_verified", when they had no usage of emails
      # This can be overridden with the below option
      #insecureSkipEmailVerified: true
    # Define an ID and a name to this IdP connector
    id: {connector-id}
    name: {connector-name}
    type: oidc
    getUserInfo: true
  expiry:
    idTokens: 10m
    refreshTokens:
      validIfNotUsedFor: 15m # Define a value which is no shorter than 15 minutes (15m) and no l
      absoluteLifetime: 24h
      disableRotation: false
      reuseInterval: 1m
  # Register a UI application (DM Edge or your own custom UI application) with dex as a client
  staticClients:
  - id: {dex-client-id} # Define an ID for the UI application
    secret: {dex-client-secret} # Define a secure secret for the UI application
    name: '{dex-client-name}' # Define a meaningful name for the UI application
    # Callback URL for Dex to redirect the user post authentication
    redirectURIs:
      - https://edge-ui.{dm-edge-hostname}/callback
ingress:
  enabled: true
```

```
    className: nginx
    hosts:
      - host: {dex-server-url}
        paths:
          - path: /
            pathType: Prefix
    tls:
      - hosts:
          - {dex-server-url}
        # Specify the secret you created previously for the Dex server certificate
        secretName: {dex-server-certificate}
```

Install Dex

The following procedure describes how to install a Dex server on Kubernetes, with configurations described above:

1. Add a chart repository, for example, “dex”:

   ```
   helm repo add dex https://charts.dexidp.io
   ```

2. Update the repository:

   ```
   helm repo update
   ```

3. Check the latest version for installation:

   ```
   helm search repo dex
   ```

4. Set the Kubernetes context:

   ```
   export KUBECONFIG=[kubeconfig_file_path]
   ```

5. Install Dex:

   ```
   helm install [release_name] dex/dex --version [helm_chart_version] --values [values_file] -n
   ```

Configure SAP Digital Manufacturing for edge computing

Procedure

1. Create a secret to store the signing CA certificate for the Dex server certificate in the SAP Digital Manufacturing for edge computing namespace:

   ```
   kubectl create secret generic [dex-ca-secret-name] --from-file=ca.crt=[Dex_CA_Certificate] -n
   ```

2. When installing or upgrading SAP Digital Manufacturing for edge computing, configure these values in the values file for Dex integration and for the IdP user groups:

   ```
   global:
     dex:
       useEmbedded: false
     dexIssuer: 'https://dex.example.com'
     dexCACertificate: [dex-ca-secret-name]
     dexClient:
       clientID: ***
       clientSecret: ***
     dexClientSecretName: 'dex-client-secret'
     idpLogoutUrl: 'https://my-idp-domain/oauth2/logout?post_logout_redirect_uri=https://edge-ui
   ```

```
amsAuthFramework:
  supervisorGroupName: 'supervisor-user-group'
  operatorGroupName: 'operator-user-group'
```

> **Note**
>
> If you're using an external secret, you no longer need the client ID and client secret.

Related Information

Helm Values

Use Embedded Dex

Unlike an external Dex that you install yourself, the embedded Dex is a Dex server provided by SAP Digital Manufacturing for edge computing right out of the box.

Related Information

Configure Dex as Client Application in IdP
Configure User Groups in IdP
Configure SAP Digital Manufacturing for edge computing

Configure Dex as Client Application in IdP

As Dex needs to request identity from an upstream IdP, you need to configure the following information about the Dex server in the IdP:

- Redirect or callback URI: this should be in the pattern of `https://edge-ui.<dm-edge-hostname>/dex-server/callback`
- Connector type: OIDC

  > **Note**
  >
  > The OIDC issuer URL must contain "https://". Dex does not support legacy or insecure issuer URLs.

- Client authentication method: client ID and secret for authentication. Refer to specific SAP Digital Manufacturing for edge computing values (`dex.connector.config.clientID`, `dex.connector.config.clientSecret`) in Helm Values.
- Ensure that your IdP sends all required OpenID Connect claims in addition to these additional claims:
  - email (string)
  - groups (list of strings)

  You may use the OpenID Connect Discovery endpoint to verify the claims supported by your IdP: `<issuer-url>/.well-known/openid-configuration`

  If the supported claims do not contain the required claims, you need to work with your IdP administrator to configure custom claims or claim mappings.
- Grant type: Authorization code, Refresh

> **Note**
>
> PKCE is not supported by Dex.

- Post logout redirect URL: this should be in the pattern of `https://edge-ui.<dm-edge-hostname>`. Refer to specific SAP Digital Manufacturing for edge computing values (`global.idpLogoutUrl`) in Helm Values.
- Refresh token and access token

  We recommend setting the refresh token validity of your IdP or Dex ID token to under 24 hours. The shorter time from the two values is used as the session time for each login of the edge UI. The maximum session time is 24 hours. However, you need to stay active during the session. If you're inactive for over 15 minutes, you'll be logged out.

  We suggest setting a shorter access token validity, like one hour.

Refer to the documentation of your IdP on how to configure service providers.

Configure User Groups in IdP

SAP Digital Manufacturing for edge computing supports two user groups with different scopes, targeted at supervisors and operators respectively. For details on permissions each authorization policy offers to apps on the edge landing page, see Authorization Policy.

You need to map these authorization policies directly to user groups defined in the IdP.

In your IdP:

1. Configure one or two user groups for your users to access UI applications. For details, see Create User Groups.
2. Assign users to the appropriate user group. For details, see Import Users for SAP Digital Manufacturing.

When installing SAP Digital Manufacturing for edge computing, define your user groups in the `global.security.amsAuthFramework.supervisorGroupName` and `global.security.amsAuthFramework.operatorGroupName` values. The values should match the user group names you've defined in the IdP.

> **→ Tip**
>
> You are recommended to configure different groups for different edge devices.

Configure SAP Digital Manufacturing for edge computing

Procedure

1. Create a secret to store the signing CA certificate for the SAP Digital Manufacturing for edge computing server certificate in the SAP Digital Manufacturing for edge computing namespace. For embedded Dex, the Dex server certificate is the SAP Digital Manufacturing for edge computing server certificate.

   ```
   kubectl create secret generic [dex-ca-secret-name] --from-file=ca.crt=[Dex_CA_Certificate] -n
   ```

2. When installing or upgrading SAP Digital Manufacturing for edge computing, configure these values in the values file for Dex integration and for the IdP user groups:

   ```
   dex:
     dexConnectorSecretName: 'dex-connector-secret'
   ```

```
    connector:
      issuer: 'https://my-idp-domain'
      config:
        clientID: ***
        clientSecret: ***
      userIDKey: 'sub'
  global:
    dex:
      useEmbedded: true
    dexCACertificate: [dex-ca-secret-name]
    dexClient:
      clientID: ***
      clientSecret: ***
    dexClientSecretName: 'dex-client-secret'
    idpLogoutUrl: 'https://my-idp-domain/oauth2/logout?post_logout_redirect_uri=https://edge-ui
    security:
      amsAuthFramework:
        supervisorGroupName: 'supervisor-user-group'
        operatorGroupName: 'operator-user-group'
```

Note

If you're using an external secret, you no longer need the two pairs of client IDs and client secrets.

Related Information

Helm Values

Upgrade with Helm

If you want to apply a new version of SAP Digital Manufacturing for edge computing or apply some additional configurations like ingress TLS/SSL certificates, you need to upgrade the application.

Prerequisites

- Check if you have sufficient resources for the rolling update. Leave at least 2.5 CPU and 16 GB memory available to use.
- If you need to make module changes such as deploying a new module or un-deploying an existing module, edit the edge device and obtain the updated configuration information regarding "tags" values first in the **Manage Edge Devices** app, and modify the values file accordingly. Additionally, if you use a local container registry, to deploy a new module, be sure to pull the images for this module into the local container registry. See Get Docker Image List for details.

Procedure

1. Set the Kubernetes context:

   ```
   export KUBECONFIG=[kubeconfig_file_path]
   ```

   Note

   For Windows, the command is `set KUBECONFIG=[kubeconfig_file_path]`.

2. Update the local Helm repository for SAP Digital Manufacturing for edge computing:

   ```
   helm repo update [helm_repo]
   ```

   Note

If you pull images directly from the RBSC, you may need to remove the existing Helm repository and add it again using the new credentials because the RBSC technical user expires every six months.

3. Find out the exact installed release in the SAP Digital Manufacturing for edge computing's namespace:

```
helm list -n [namespace] --all
```

4. Update the values file as necessary.
5. Find out the available upgradable version:

```
helm search repo [helm_repo]/offline-edge
```

6. Upgrade the application:

```
helm upgrade [release_name] -f [values_file] [helm_repo]/offline-edge -n [namespace] --versio
```

> **Note**
> If you are testing beta versions, which are reserved for the quality testing window before official release, add the "--devel" option.

Next Steps

An initial load for an upgrade is necessary when deploying new modules or performing major version upgrades. Check in the **Manage Edge Devices** app whether an initial load is required, then trigger an initial load to synchronize data from the cloud to the edge in the **Initial Load** tab of the app. For details, see Data Load and Enable and Trigger an Initial Load.

Uninstall with Helm

Prerequisites

You have ensured that all your edge transactions have been delivered to the cloud. Use `/api/edge/transactions/v1/transactions` API to check this. See Get Overview of Edge Transactions for details.

Procedure

1. Set the Kubernetes context:

```
export KUBECONFIG=[kubeconfig_file_path]
```

> **Note**
> For Windows, the command is `set KUBECONFIG=[kubeconfig_file_path]`.

2. Find out the exact installed release in the namespace for SAP Digital Manufacturing for edge computing:

```
helm list -n [namespace] --all
```

3. Uninstall the application:

```
helm uninstall [release_name] -n [namespace]
```

4. Check if the edge device is deleted from the Manage Edge Devices app. If it's not, delete it. For details, see Delete Edge Device.

   In the uninstallation process, all pods and config maps are deleted from the SAP Digital Manufacturing for edge computing namespace, except for the dm-edge-common-configmap. You can use the dm-edge-common-configmap to verify if the edge devices are deleted from the cloud. After verification, delete the dm-edge-common-configmap.

5. Delete the entire SAP Digital Manufacturing for edge computing namespace.

   ```
   kubectl delete namespace [namespace]
   ```

6. If you use Istio as the ingress controller, delete the Kubernetes secrets storing trusted CAs from the Istio namespace.

   ℹ Note

   Uninstallation is prevented if there are still pending transactions to be relayed to the cloud. However, if you would like to uninstall anyway, set `waitTransactions` in the values.yaml file to `false` to lift this block. As pending transactions will be lost in this process, it is advisable to uninstall after the transactions have arrived at the cloud to be processed. Remember to set `waitTransactions` back to `true` when you perform the uninstallation later.

   ⚠ Caution

   If you use Argo CD, which does not support pre-delete hook, uninstallation won't be blocked even if this option is set to true. In this case, use the API `/api/edge/transactions/v1/transactions` (see Get Overview of Edge Transactions) to check the pending transactions before you delete the application.

   If you want to create an edge device with the same plant after uninstallation, but your edge device remains registered on the cloud due to any reason (for example, incomplete uninstallation), try to delete the edge device from the cloud. For details, see Delete Edge Device.

   When working with external databases, you need to use a dedicated one specifically for SAP Digital Manufacturing for edge computing. After uninstallation, you need to delete the schemas from the database yourself. All the SAP Digital Manufacturing for edge computing schemas start with "sap_de_".

   Here is an example SQL query for PostgreSQL: `DROP SCHEMA IF EXISTS sap_de_core CASCADE;`

   For Microsoft SQL Server, some tables are also created in the default database owner (dbo) schema. These tables start with "ACT_", "FLW_", "PEMON__", "PE_", or "EVENT_". You also need to clean up the table for the Production Connectivity Model: "SAP.DMI.MACHINEMODEL::".

   This is also a prerequisite if you reuse the same database for another SAP Digital Manufacturing for edge computing installation.

Delete Edge Device

If your deployment tool does not support Helm pre-delete hooks (for example, some versions of Argo CD) or the pre-delete jobs run into errors, your edge device may still remain registered on the cloud while your edge installation has already been un-deployed. This will block the plant from being associated with a new device. In this case, you need to delete the edge device from the cloud.

Prerequisites

You have been assigned the role of `Edge_Admin`.

Context

Use the DEVICE_ID data in dm-edge-common-configmap to find out which edge device remain registered on the cloud.

Procedure

1. Go to the **Manage Edge Devices** app.
2. Choose the device you want to delete from the list page.
3. On the detail page, choose **Delete**.
4. For devices in the **Awaiting Installation** status, confirm your device name and delete it.

   For devices in the **Installed** status, make sure that there are no pending transactions. Double confirm by typing the edge device name and choose **Delete**.

   > ⚠ **Caution**
   >
   > This action is not reversible.

Results

The device is unregistered from the associated plant and deleted from the cloud.

Helm Values

Only one replica is supported.

Parameter	Description
`proxySetting.httpProxy`	URL for your HTTP proxy, with the port number. If it requires user authentication, enter it in this forma `http://userName:yourPassword@yourProxyl`
`proxySetting.httpsProxy`	URL for your HTTPS proxy, with the port number. If it requires user authentication, enter it in this forma `http://userName:yourPassword@yourProxyl`
`proxySetting.noProxy`	Add the addresses for which you want to bypass the p separate the values. → **Tip** We recommend keeping the default values `.svc,` `.` external Dex, add the domain of the Dex server as we
`tags`	Use this parameter to specify which modules you wou Supported modules include: • MQTT (tag: mqtt) ℹ **Note** This module is to be deprecated along with t Broker in release 2602. You are recommende MQTT brokers instead. • Production Process Orchestration (tag: proces

Parameter	Description

	• Production Connectivity Model (tag: productic
	• Production Connector (tag: production-conne
	• Printing (tag: printing)
	• POD (tag: pod)
	If you would like to enable the POD module, for examp
	Sample Code
	`tags:`
	`  pod: true`

`deviceId`	Edge device ID, obtained from cloud for the edge devic Devices app
`connectionString`	A string used for cloud–edge connection, obtained fro device in the Manage Edge Devices app
`dex.connector.issuer`	Issuer URL that identifies the IdP as an OpenID Conne Configurations for integrating the embedded Dex with provider (IdP). For an external Dex, configure the conn upstream IdP in Dex by yourself.
`dex.connector.config.clientID`	ID and secret of OpenID Connect client that you confi represent the embedded Dex Configurations for integrating the embedded Dex with provider (IdP). For an external Dex, configure the conn upstream IdP in Dex by yourself. **Note** If you're using an external secret, the system ignores you should define these values in your external secre system.
`dex.connector.config.clientSecret`	
`dex.connector.userIDKey`	The set claim of user ID defined in your organization's consistent with the user ID in the Manage User Assign
`dex.dexConnectorSecretName`	Kubernetes secret which stores the connector inform
`global.ingressType`	Value must be either “nginx” or “istio”. **Note** The value is case-sensitive.
`global.ingressClassName`	Specifies the ingress controller which handles traffic t Manufacturing for edge computing.
`global.ingressCACertificate`	Kubernetes secret for storing trusted CA certificates f

Parameter	Description
	1. Signing CA for the SAP Digital Manufacturing client certificate for calling SAP Digital Manufa computing APIs 2. Signing CA of the Cloud Connector system ce For details, see Install with Helm.
`global.ingressServerCertificate`	Kubernetes secret for storing the SAP Digital Manufac computing server certificate. For details, see Install with Helm.
`global.nginxNamespace`	Namespace where NGINX ingress is installed
`global.istioNamespace`	Namespace where Istio is installed
`global.tenantId`	Your SAP Digital Manufacturing BTP subaccount ID, o the edge device in the **Manage Edge Devices** app
`global.dockerRegistry`	URL of container registry. If you have pulled the docke container registry, you need to specify the URL manua
`global.storageClass`	Storage class of the persistent volumes. Unless explic storage class will be used. If you haven't configured ar default storage class is set, the installation will fail. Use this command to check the storage classes: `kube storageclass`
`global.gateway.hostname`	The hostname mapped to the external IP address of t UI needs to have the same domain as this hostname.
`global.uaaClientId`	UAA client ID, obtained from the service key of an SAP service instance in your BTP subaccount
`global.uaaClientSecret`	UAA client secret, obtained from the service key of an Manufacturing service instance in your BTP subaccou
`global.uaaUrl`	UAA URL, obtained from the service key of an SAP Dig service instance in your BTP subaccount **Note** Print service reuses this global value for UAA URL.
`global.uaaVerificationKey`	UAA verification key, obtained from the service key of Manufacturing service instance in your BTP subaccou This public key is used for token verification on incom configuration APIs of SAP Digital Manufacturing for ec
`global.serviceKeySecretName`	Kubernetes secret for UAA client ID, client secret, URL
`global.dex.useEmbedded`	Keep it set to false if you want to use your own Dex. Se use the embedded Dex.

Parameter	Description
`global.dexIssuer`	URL to OIDC token issuer, also known as the Dex host authenticating users for UI access. **Note** The OIDC issuer URL must contain "https://". Dex do or insecure issuer URLs.
`global.dexCACertificate`	Kubernetes secret for the CA certificate for Dex. Appli and embedded Dex. For embedded Dex, the SAP Digital Manufacturing for certificate is the Dex server certificate. For details, see Install with Helm.
`global.dexClient.clientId`	Client ID and secret configured in your Dex server for
`global.dexClient.clientSecret`	
`global.dexClientSecretName`	Kubernetes secret which stores the information of the
`global.idpLogoutUrl`	Set `idpLogoutUrl` to log out from the upstream IdP configured, the user will remain logged in. Include `post_logout_redirect_uri` and `client_id` as redirection after logout. These two values are configur an OpenId Connect client of the IdP. For example, `htt domain>/oauth2/logout?` `post_logout_redirect_uri=https://edge-` `hostname>&client_id=<Client Id>`
`global.database.useEmbedded`	Set to true if you want to use embedded PostgreSQL
`global.database.type`	Choose either `postgresql` or `mssqlserver` if you
`global.database.auth.host`	Host of an external database.
`global.database.auth.port`	Port of an external database.
`global.database.auth.database`	Name of an external database.
`global.database.auth.username`	Username of an external database. **Note** The database user must have all privileges including privilege granted on the given database.
`global.database.auth.password`	This value will be ignored if you use an external secret Specify this parameter value to use your own passwor generated password for the database. For details, see Password for Embedded Database.
`global.imagePullSecret`	Kubernetes secret for the RBSC docker registry or you registry. It is automatically created if you use external

Parameter	Description
`global.security.imageCredentials.username`	The user name and password for the RBSC docker reg container registry. If you directly pull images from the RBSC, use the tech your S-user account.
`global.security.imageCredentials.password`	
`global.security.externalSecrets.enabled`	Enable this setting to use an external secret for the se database password, Dex information and docker regis details, see Use External Secrets for Credentials. **Note** To use an external secret for the service key UAA val parameter and `global.security.externalSecrets.servi` should be enabled. To use an external secret for database password, bo `global.security.externalSecrets.datab` be enabled. To use an external secret for Dex information, both t `global.security.externalSecrets.dex.e` enabled. To use an external secret for docker registry credent parameter and `global.security.externalSecrets.docke` should be enabled.
`global.security.externalSecrets.serviceKey.enabled`	Enable this setting to use an external secret for the se details, see Use External Secrets for Credentials.
`global.security.externalSecrets.serviceKey.remoteRefKey`	Secret created in the external secret management for credentials.
`global.security.externalSecrets.dex.enabled`	Enable this setting to use an external secret for Dex in see Use External Secrets for Credentials.
`global.security.externalSecrets.dex.remoteRefKey`	Secret created in the external secret management for credentials.
`global.security.externalSecrets.dockerRegistry.enabled`	Enable this setting to use an external secret for docke details, see Use External Secrets for Credentials.
`global.security.externalSecrets.dockerRegistry.remoteRefKey`	Secret created in the external secret management for credentials.
`global.security.externalSecrets.database.enabled`	Enable an external secret for the database password. External Secrets for Credentials.
`global.security.externalSecrets.database.remoteRefKey`	Secret created in the external secret management for database passwords credentials.
`global.security.externalSecrets.secretStoreRefKind`	Secret store kind. For details, see Use External Secret

Parameter	Description
`global.security.externalSecrets.secretStoreRefName`	Secret store name. For details, see Use External Secre
`global.security.externalSecrets.remoteRefKey`	Secret created in the external secret management for user-defined database passwords, Dex information an credentials. For details, see Use External Secrets for C
`global.security.amsAuthFramework.supervisorGroupName`	A user group in your IdP for production supervisors
`global.security.amsAuthFramework.operatorGroupName`	A user group in your IdP for production operators
`global.customServiceNamespace`	If you intend to develop a custom service that is authe policy and service account tokens for invoking SAP Di edge computing APIs, specify a namespace for this se must be in the same cluster as SAP Digital Manufactu
`global.customServiceAccount`	Service account of custom service for service account a name for the service account your custom services with SAP Digital Manufacturing for edge computing.
`edge-mqtt-broker.brokerconf.tokenVerificationKey`	**Note** Deprecated. Use `global.uaaVerificationKey` instead.
`edge-mqtt-broker.installBrokerCRDs`	This parameter controls the deployment of Custom R (CRDs) required to deploy the SAP MQTT Broker. See CRDs" on Deployment of SAP Digital Manufacturing fo additional details.
`edge-mqtt-broker.nginx.installConfigMap`	This parameter controls the deployment of a ConfigM namespace, including a configuration parameter that maximum header size accepted by the NGINX Ingress "Install ConfigMap with NGINX configuration" on Depl Manufacturing for Edge Computing for additional deta
`print-service.dataSync.serviceUrl`	URL of Cloud Print Service for data synchronization, fr the Print Service instance
`print-service.dataSync.uaaClientId`	UAA ClientID from the service key of the Print Service
`print-service.dataSync.uaaClientSecret`	UAA ClientSecret from the service key of the Print Ser
`waitTransaction`	For uninstallation only. By default, uninstallation is prevented if there are still be relayed to the cloud. If you would like to uninstall anyway, for example, in a this parameter to `false`. These pending transactions **Caution** If you use Argo CD, which does not support pre-dele won't be blocked even if this option is set to true. In t `/api/edge/transactions/v1/transaction` Edge Transactions) to check the pending transactior application.
`edge-object-store.storageClass`	Storage class of the persistent volumes of the `edge-`

Parameter	Description
`edge-object-store.hasRWXSupport`	Specify if the storage class for the `edge-object-st` the "ReadWriteMany" (RWX) access mode.
`dm-prodcon.storageClass`	Storage class of the persistent volumes of the Produc **Note** This storage class must support RWX.
`dm-prodcon.listOfAdminUsers`	The list of administrator users separated by semicolo `production-connector` module.

Related Information

Example of Helm Values File

Example of Helm Values File

Below is a sample `values.yaml` file with basic configurations.

Example

Sample Code

```
proxySetting:
  httpProxy: 'http://username:password@hostname:port'
  httpsProxy: 'http://username:password@hostname:port'
  noProxy: '.svc,.local'
waitTransaction: true
deviceId: '2acbd363-05e8-42b3-8d-example-device'
tags:
  production-connectivity: false
  production-connector: false
  process-orchestration: false
  printing: true
  pod: false
connectionString: 'KAFKA_ORG=az-quality;DOMAIN=dm.eu20-quality.prod.shoot.live.k8s-hana.ondemand
edge-object-store:
  storageClass: 'my-storage-class'
  hasRWXSupport: true
dm-prodcon:
  storageClass: 'my-storage-class-2'
  listOfAdminUsers: best.run1@example.com;best.run2@example.com
print-service:
  dataSync:
    serviceUrl: 'https://api.eu20.print.services.sap'
    uaaClientId: 'my-print-uaa-client-id'
    uaaClientSecret: 'my-print-uaa-client-secret'
dex:
  dexConnectorSecretName: 'dex-connector-secret'
  connector:  # Relevant because embedded Dex is being used
```

```
      issuer: 'https://my-idp-domain'
      config:
        clientID: 'my-idp-oidc-client-id'
        clientSecret: 'my-idp-oidc-client-secret'
  global:
    tenantId: '0982a0-b46b-4b3e-9740-example-tenant'
    dockerRegistry: '73555000100900006833.dockersrv.base.repositories.cloud.sap'
    storageClass: default
    gateway:
      hostname: 'my-edge-host.example-domain'
    uaaClientId: 'my-dm-uaa-client-id'
    uaaClientSecret: 'my-dm-uaa-client-secret'
    uaaUrl: 'https://my-subdomain.authentication.eu20.hana.ondemand.com'
    uaaVerificationKey: 'my-dm-uaa-verification-key'
    serviceKeySecretName: 'dm-edge-service-key-secret'
    ingressType: 'nginx'
    ingressCACertificate: 'sap-dm-edge-ca-secret'
    ingressServerCertificate: 'sap-dm-edge-tls-secret'
    ingressClassName: 'nginx'
    nginxNamespace: 'kube-system'
    istioNamespace: 'istio-system'  # Irrelevant because nginx is being used
    dex:
      useEmbedded: true
    dexIssuer: 'https://dex.example.com'  # Irrelevant because embedded Dex is being used
    dexCACertificate: 'dex-ca-crt-secret'
    dexClient:  # If you intend to use the embedded Dex for your own custom UI applications, be su
      clientID: 'my-dex-client-id'
      clientSecret: 'my-dex-client-secret'
    dexClientSecretName: 'my-dex-client-secret'
    idpLogoutUrl: 'https://my-idp-domain/oauth2/logout?post_logout_redirect_uri=https://edge-ui.my
    database:
      useEmbedded: false
      type: postgresql
      auth:
        host: 'postgres'
        port: 5432
        database: 'dm-edge'
        username: 'dm-edge'
        password: 'my-database-password'
    security:
      imageCredentials:
        username: 'my-docker-registry-username'
        password: 'my-docker-registry-password'
      externalSecrets: # No external secret management system is being used
        enabled: false
        database:
          enabled: false
          remoteRefKey: null
        dockerRegistry:
          enabled: false
          remoteRefKey: null
        serviceKey:
          enabled: false
          remoteRefKey: null
```

```
        dex:
          enabled: false
          remoteRefKey: null
        secretStoreRefKind: SecretStore
        secretStoreRefName: null
        remoteRefKey: null
      amsAuthFramework:
        supervisorGroupName: 'dmedge_supervisor'
        operatorGroupName: 'dmedge_operator'
    customServiceNamespace: 'my-custom-service-namespace'
    customServiceAccount: 'my-custom-service-service-account'
  edge-mqtt-broker: # Irrelevant because MQTT module is not to be deployed
    brokerconf:
      tokenVerificationKey: "-----BEGIN PUBLIC KEY-----dm-uaa-public-key-content-----END PUBLIC KE
    nginx:
      installConfigMap: false
    installBrokerCRDs: true
```

Ways to Configure Kubernetes Secrets and Priorities

Some sensitive information like UAA client secrets is stored in Kubernetes secrets. These secrets can be provisioned in two different ways.

1. Pass the Helm values directly in the configuration file or as part of the Helm install or upgrade command. The SAP Digital Manufacturing for edge computing applications then provisions corresponding Kubernetes secrets with these values.
2. Use an external secret management system to store these values and integrate the external secret management system with your Kubernetes cluster. The integration provisions the Kubernetes secrets with these values during Helm install or upgrade. This method of provisioning takes higher priority. If external secrets are enabled, Helm values passed directly are ignored.

For some of these Kubernetes secrets, you can customize the secret names. For those that don't allow customization, a default value is used.

Below is an overview of the relevant Helm values.

Kubernetes Secret Name Helm Value	**Helm Value**	**External Secret Enablement**	**R**
global.serviceKeySecretName	global.uaaClientId	global.security.externalSecrets.enabled + global.security.externalSecrets.serviceKey.enabled	g
	global.uaaClientSecret		
	global.uaaUrl		
	global.uaaVerificationKey		
dex.dexConnectorSecretName	dex.connector.config.clientID	global.security.externalSecrets.enabled + global.security.externalSecrets.dex.enabled	g
	dex.connector.config.clientSecret		
global.dexClientSecretName	global.dexClient.clientID	global.security.externalSecrets.enabled	
	global.dexClient.clientSecret		

Kubernetes Secret Name Helm Value	Helm Value	External Secret Enablement	R
/	global.database.auth.host	global.security.externalSecrets.enabled + global.security.externalSecrets.database.enabled	g
	global.database.auth.port		
	global.database.auth.database		
	global.database.auth.username		
	global.database.auth.password		
global.imagePullSecret	global.security.imageCredentials.username	global.security.externalSecrets.enabled + global.security.externalSecrets.dockerRegistry.enabled	g
	global.security.imageCredentials.password		

Get Docker Image List

You may want to get the image list before you start Helm chart deployment. Or you may want to get the image list for an exisiting deployment.

Context

Below is an example using the Helm plugin “Helm Images”. For more information, see https://github.com/nikhilsbhat/helm-images . You may use any other tools to achieve the same purpose.

Procedure

1. Create or update your local Helm repository for SAP Digital Manufacturing for edge computing. See Install with Helm or Upgrade with Helm for related commands.
2. Change to the directory where you want to pull the Helm chart and then pull the Helm chart version, for example, 1.2402.0, from your local helm repository named as "rbsc":

   ```
   helm pull rbsc/offline-edge --version 1.2402.0
   ```

3. Find out the exact name of the corresponding tgz file:

   ```
   ls *.tgz
   ```

   Note

   The response for this command is "offline-edge-1.2402.0.tgz".

4. Get the image list from the chart with the following command and additional flags:

   ```
   helm images get offline-edge ./offline-edge-1.2402.0.tgz --unique --set global.tenantId=test
   ```

Additional Flag	Command
Module	`--set tags.printing=true --set tags.process-orchestration=true --set tags.production-conr` **Note**

Additional Flag	Command
	Refer to the tag information in the module table in Modular Deployment for exact module tag names.
Embedded Dex	`--set global.dex.useEmbedded=true --set dex.connector.config.clientID=test --set dex.conn`
Microsoft SQL Server Database	`--set global.database.useEmbedded=false --set global.database.type=mssqlserver`

i Note

To get the image list of an existing deployment (for example, `sap-dm-edge-helm`) from a namespace (for example, `sap-dm-edge`), set the Kubernetes context first and then execute this command:

```
helm images get sap-dm-edge-helm --from-release -n sap-dm-edge --unique
```

Additional MQTT Broker Images

Context

Some of the container images required to run the SAP MQTT Broker cannot be fetched using the procedure described above. These images are:

- MQTT Broker init container image (`com.sap.dm/edge-mqtt-broker-init-container`)
- MQTT Broker init container base image (`artemiscloud/activemq-artemis-broker-init`)
- MQTT Broker container image (`artemiscloud/activemq-artemis-broker-kubernetes`)

You can pull them using a Docker client ("docker" command from the command line) following the procedure described below for each of the images. You will also need to use the "helm" and the "tar" commands. Additionally, you may rely on command "yq" for querying values in a values.yaml file.

Procedure

1. Pull the Helm chart version:

   ```
   helm pull rbsc/offline-edge --version <chart-version>
   ```

2. Login to the image repository via your RBSC credentials (provide your password when prompted):

   ```
   docker login <repository-url>
   ```

3. Open the chart archive file:

   ```
   tar -xzf offline-edge-<version>.tgz
   ```

4. Query image name and tag for the specific image via the "yq" command, using the query strings in the table below. In case the "yq" command is not available, open the offline-edge/charts/edge-mqtt-broker/values.yaml file via a text editor and identify the appropriate tag in the YAML file:

```
yq 'query-string' ./offline-edge/charts/edge-mqtt-broker/values.yaml
```

Image	Query string / tag in values.yaml file
MQTT Broker init container image	`.artemis.initbroker`
MQTT Broker init container base image	`.artemis.activemqinitbroker`
MQTT Broker container image	`.artemis.broker`

5. Note down the values for "imageName" and "imageTag" for each image, then pull the container image. Do not use the complete repository URL, but only the repository hostname (remove the protocol identifier, like "http", from the complete URL):

   ```
   docker pull <repository-hostname>/<imageName>:<imageTag>
   ```

6. You can check that all of the container images were pulled correctly via the command:

   ```
   docker images
   ```

Enable Deploying Edge Processes from Cloud to Edge

This topic describes the steps needed to enable deploying edge processes, as well as event-type business rules from the cloud to the edge.

Prerequisites

- You have installed the Cloud Connector in your enterprise network. For details, see Prerequisites for Installing the Software Components and Installing the Cloud Connector.
- You have completed all the initial configurations to get your Cloud Connector up and running. This includes adding the SAP Business Technology Platform (BTP) subaccount for your SAP Digital Manufacturing tenant to the Cloud Connector. For details, see Defining the Subaccount.

  > **Note**
  >
  > You have also obtained a Cloud Connector system certificate. Self-signed system certificate is not supported. The system certificate must use the domain name, preferably FQDN (fully qualified domain name), as the common name. For details, see Create a CSR and Import the Signed Certificate.

- You have obtained the CA certificate signing the SAP Digital Manufacturing for edge computing server certificate. The CA certificate file must include the entire CA chain, starting with the signing CA and ending with the root CA.
- You have installed SAP Digital Manufacturing for edge computing and added the SAP Digital Manufacturing for edge computing server certificate as well as the signing CA for the Cloud Connector system certificate, including its root CA, in the respective Kubernetes secrets. For details, see Install with Helm.
- You have deployed the **Production Process Orchestration** module when installing the edge. See Modular Deployment.

Procedure

1. Access the Cloud Connector web interface and on the **Cloud Connector Administration** screen, choose **Configuration** > **On-Premises tab** > **Backend Trust Store section**, switch on the **Determining Trust Through Allow List** toggle, upload the CA certificate signing the SAP Digital Manufacturing for edge computing server certificate to the allow list.

2. Choose **Select Subaccount** to select the subaccount you have added and select **Cloud to On-Premises** on the left panel.
3. Choose **Access Control tab** > **Mapping Virtual to Internal System** and choose □ *(Add)* to add the system mapping. Enter the following data:

Property	Value
Back-end Type	Other SAP System
Protocol	HTTPS
Internal Host	Enter the hostname of the edge device.
Internal Port	443
Virtual Host	Enter the virtual host. This is a unique name for the edge instance and is used for its identification.
Virtual Port	Enter the virtual port, for example, 50000. **i Note** Internal port and virtual port can be the same.
Principal Type	Select the **None** option.
System Certificate for Logon	Select the checkbox.
Host In Request Header	Use Internal Host
Description	Not required.
Check Internal Host	Select the checkbox.

4. Choose **Finish**.
5. Select the system mapping you have just added. Choose □ *(check availability of internal host)* to ensure the connection to the Cloud Connector has been successfully established.
6. In the **Resource Of [virtual host]** section, choose □ *(Add)* to add the resources of the edge instance that SAP Digital Manufacturing can access via the Cloud Connector. Enter the following data and choose **Save**:

Property	Value
URL Path	/processengine
Active	Select the checkbox.
WebSocket	De-select the checkbox.
Access Policy	Path And All Sub-Paths
Description	Not required.

7. To enable deploying event-type business rules to the edge, enter the following data:

Property	Value
URL Path	/businessrule/process
Active	Select the checkbox.

Property	Value
WebSocket	De-select the checkbox.
Access Policy	Path And All Sub-Paths
Description	Not required.

8. In the **Configure Production Connectivity** app of SAP Digital Manufacturing launchpad, add an entry for your Cloud Connector, if not already done. For details, see Creating a Cloud Connector Object.
9. In the **Manage Web Servers** app, choose the corresponding SAP Digital Manufacturing for edge computing services type web server `DM_Edge_<EdgeDevice>`.

   The web server is automatically created on successful edge device installation.
10. Choose **Edit**.
11. Add the Cloud Connector and the virtual host URL for your edge instance. Make sure to include “http://” and the port number in the virtual host URL.
12. Choose **Save**.

Next Steps

You can create edge processes using your edge web server as the runtime. Test whether you can deploy them successfully. You can apply a similar test for business rules with edge events on your edge web server.

Related Information

Available Services and Subprocesses
Manage Subscriptions
Enable Production Connector to Invoke Edge Services
Enable Deploying Automation Sequences of Production Connector on edge Runtime

Enable Deploying Automation Sequences of Production Connector on edge Runtime

This topic describes the steps needed to deploy the automation sequence or subscription of the Production Connector on edge runtime web server.

Prerequisites

- You have installed the Cloud Connector in your enterprise network. For details, see Prerequisites for Installing the Software Components and Installing the Cloud Connector.
- You have completed all the initial configurations to get your Cloud Connector up and running. This includes adding the SAP Business Technology Platform (BTP) subaccount for your SAP Digital Manufacturing tenant to the Cloud Connector. For details, see Defining the Subaccount.

  > **Note**
  >
  > You have also obtained a Cloud Connector system certificate. Self-signed system certificate is not supported. The system certificate must use the domain name, preferably FQDN (fully qualified domain name), as the common name. For details, see Create a CSR and Import the Signed Certificate.

- You have obtained the CA certificate signing the SAP Digital Manufacturing for edge computing server certificate. The CA certificate file must include the entire CA chain, starting with the signing CA and ending with the root CA.
- You have installed SAP Digital Manufacturing for edge computing and added the SAP Digital Manufacturing for edge computing server certificate as well as the signing CA for the client certificate, including its root CA, in the respective Kubernetes secrets. For details, see Install with Helm.
- You have deployed the **Production Process Orchestration**, **Production Connectivity Model**, and **Production Connector** module when installing the edge. See Modular Deployment.
- You have added the system mapping in the Cloud Connector described in Enable Deploying Edge Processes from Cloud to Edge.

Procedure

1. Access the Cloud Connector web interface and on the **Cloud Connector Administration** screen, choose **Configuration** > **On-Premises tab** > **Backend Trust Store section** >, switch on the **Determining Trust Through Allow List** toggle, upload the CA certificate signing the SAP Digital Manufacturing for edge computing server certificate to the allow list.
2. Choose **Select Subaccount** to select the subaccount you have added and select **Cloud to On-Premises** on the left panel.
3. Choose the **Access Control tab** > **Mapping Virtual to Internal System** > and select your edge system mapping.
4. In the **Resource of [virtual host]** section, choose □ *(Add)* to add the resources of the edge instance that SAP Digital Manufacturing can access via the Cloud Connector. Enter the following data and choose **Save**:

Property	Value
URL Path	/cloudservices
Active	Select the checkbox.
WebSocket	De-select the checkbox.
Access Policy	Path And All Sub-Paths
Description	Not required.

5. Check in the **Configure Production Connectivity** app if the Production Connector object `DM_Edge_Production_Connector_<Edge_Device_Name>` has been automatically created upon edge installation with related modules deployed.
6. Maintain the required Cloud Connector and enter the Production Connector Virtual Host URL details for this object for a successful connection. Make sure to include “http://” and the port number in the virtual host URL. For details, refer to Updating a Production Connector on Edge Object.
7. Check if this object in the **Manage Web Servers** app is automatically connected to your edge web server.
8. Add the required services that you want to call either from a subscription or an automation sequence, from the Production Connector on edge. For details, refer to Adding and Deploying Services from a Connected Web Server.

Next Steps

You can create automation sequences or subscriptions with the Production Connector on edge runtime. Use some edge services in these automation sequences and test if you can deploy them. Remember, you also need to create client proxies for them. For details, see Configure Client Proxy for Third-Party Service.

Related Information

Available Services and Subprocesses

Manage Subscriptions

Authorization Policy

Find out available authorization policies and the permissions associated with each policy.

An authorization policy is a set of rules and criteria that determine what actions or operations a user or system component can perform within a system or application. It defines the permissions and access control for resources based on the identity of the user, the role they hold, or other attributes like time or location.

SAP Digital Manufacturing for edge computing has two predefined authorization policies with different scopes, targeted at supervisors and operators respectively.

The table shows the available authorization policies within SAP Digital Manufacturing for edge computing and how they are applied through the Helm values definition.

Authorization Policy	Helm Value	Description	Note
authorization policy for supervisor	`global.security.amsAuthFramework.supervisorGroupName`	The value corresponds to authorization policy for supervisors on edge.	The value is abbreviated as “Supervisor” in the following table.
authorization policy for operator	`global.security.amsAuthFramework.operatorGroupName`	The value corresponds to authorization policy for operators on edge.	The value is abbreviated as “Operator” in the following table.

SAP Digital Manufacturing for edge computing relies on Dex to delegate authentication to the upstream IdP. Users must belong to authorized user groups to access SAP Digital Manufacturing for edge computing. You need to map the authorization policies above directly to user groups defined in the IdP. For details, Configure User Groups in IdP.

Authorization Policies, Apps, and Permission

The table describes permissions each authorization policy offer to apps on the edge landing page.

App	Authorization Policy	Permissions
Cloud Connection Status	Supervisor	Assigned users can do the following: • Check the connection status between cloud and edge
Cloud Connection Status	Operator	Assigned users can do the following: • Check the connection status between cloud and edge
Monitor Production Processes	Supervisor	Assigned users can do the following: • View the execution result and progress of production processes of

App	Authorization Policy	Permissions
		edge and Production Connector on edge runtime • Terminate edge processes • View statistics report
Monitor Production Processes	Operator	Assigned users can do the following: • View the execution result and progress of production processes of edge and Production Connector on edge runtime • Terminate edge processes • View statistics report
Monitor Event Business Rules	Supervisor	Assigned users can do the following: • View the condition evaluation and action trigger status of edge event-type business rules

Appendix 3: Generate Self-Signed Certificates Using OpenSSL

For testing purposes, you can generate self-signed certificates for calling edge APIs. The example below uses the tool OpenSSL and requires it to be installed on your workstation beforehand.

Context

This example is for testing purposes only. It creates only one custom CA for both the server certificate and the client certificate. However, in productive scenarios, it may be different signing or intermediate CAs, or even different root CAs.

Procedure

1. Generate a CA key pair (ca.key) and CA certificate (ca.crt):

```
openssl req -x509 -sha256 -newkey rsa:4096 -keyout ca.key -out ca.crt -days 365 -nodes -subj
```

2. Generate a server key pair (server.key) and sign the server certificate signing request (server.csr) with the CA certificate:

    a. Create a configuration text file (server_csr.cnf), replacing the placeholder values with your own specific information:

```
[req]
distinguished_name = req_distinguished_name
prompt = no

[req_distinguished_name]
C = DE #YourCountry (in two capitals)
ST = Baden-Württemberg #YourStateOrProvince
O = SAP #YourOrganization
CN = bestrun.example.com #common name

[v3_req]
keyUsage=critical,digitalSignature,keyEncipherment
extendedKeyUsage=serverAuth
subjectAltName = @alt_names
```

```
[alt_names]
DNS.1 = bestrun.example.com
DNS.2 = edge-ui.bestrun.example.com
```

Note

The above two alternative names must be included in the file. Meanwhile, you can define more alternative names, such as IP addresses, in the file. Replace "bestrun.example.com" with your hostname.

b. Create a certificate signing request (CSR) and verify the content:

```
openssl req -new -newkey rsa:4096 -keyout server.key -out server.csr -nodes -config serv
openssl req -text -noout -verify -in server.csr
```

c. Sign CSR (server.csr) with the CA certificate and its private key:

```
openssl x509 -req -sha256 -days 365 -in server.csr -CA ca.crt -CAkey ca.key -set_serial
```

d. Inspect the certificate if alternative names are present:

```
openssl x509 -in server.crt -text -noout
```

3. Generate a client key pair (client.key) and sign the client CSR (client.csr) with the CA certificate:

```
openssl req -new -newkey rsa:4096 -keyout client.key -out client.csr -nodes -subj "/CN=dm-edg
openssl x509 -req -sha256 -days 365 -in client.csr -CA ca.crt -CAkey ca.key -set_serial 02 -o
```

Note

The common name must be `dm-edge-services` for the client certificate to integrate with SAP Digital Manufacturing for edge computing.