FUJITSU

FUJITSU Software Enterprise Service Catalog Manager V18.0.0

QuickStart Guide

April 2019

Contents

About this Manual

This manual describes how to get started with) FUJITSU Software Enterprise Service Catalog Manager, hereafter referred to as ESCM.

Readers of this Manual

This manual is directed to operators who want to quickly set up a basic installation of ESCM with Docker and Docker Compose. For more detailed information on configuration and usage, refer to the official *ESCM documentation*. It assumes that you are familiar with the following:

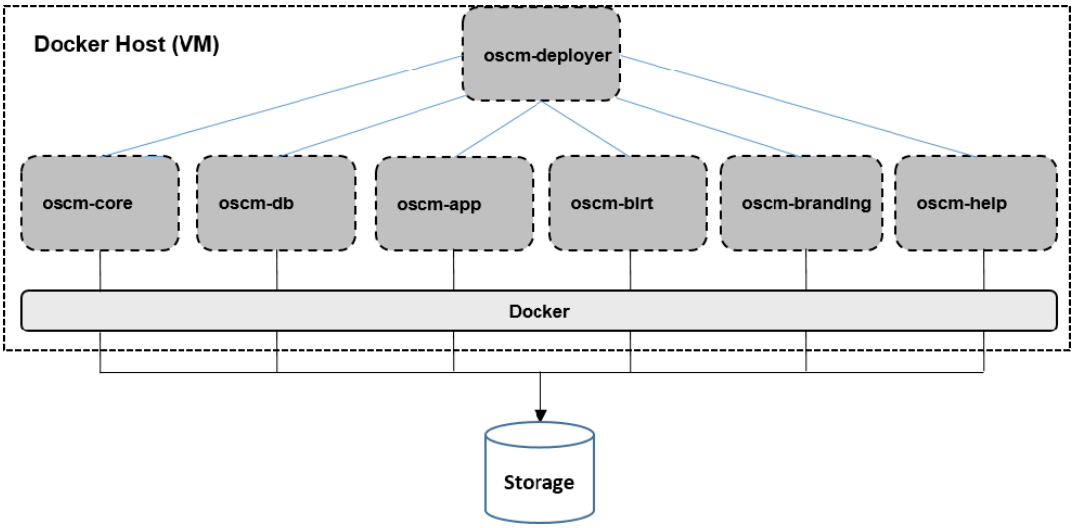
- Administration of the operating systems in use, including the adaption and execution of batch files or shell scripts.
- Java EE technology, particularly as to the deployment on application servers.
- Container technology, particularly Docker and Docker Compose.
- ESCM concepts as explained in the Overview manual.

1 ESCM Containers

ESCM is provided in Docker containers and deployed in a container environment such as OpenStack or a Kubernetes cluster providing for optimum performance, scalability, failover, and non-stop operation. The applications integrated with ESCM and their data may be hosted in the same Virtual Machine (VM) as ESCM or in different locations.

The `oscm-deployer` container is used for the configuration and deployment of the following ESCM containers:

- `oscm-core`: The ESCM core application.
- `oscm-birt`: The report engine that ESCM uses for generating reports.
- `oscm-branding`: A static Web server providing an empty directory structure for customizing the layout and branding of ESCM marketplaces.
- `oscm-help`: A static Web server providing the online help for the ESCM administration portal and marketplaces.
- `oscm-app`: The Asynchronous Provisioning Platform (APP) together with an OpenStack, an Amazon Web Services (AWS), a Microsoft Azure, a VMware, and a Shell service controller.
- `oscm-db`: Database SQL server providing the database schema required for running ESCM and APP.



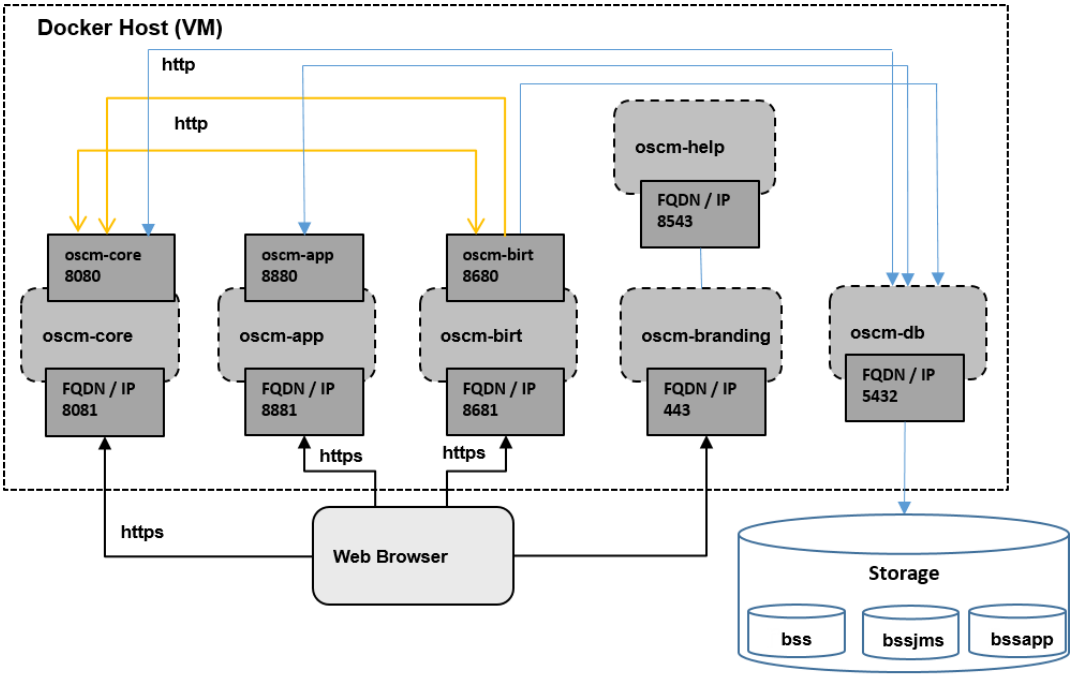
ESCM and APP store their data in PostgreSQL databases. For each database, a volume for persistent storage is mounted during the deployment process. The databases write to these volumes so that data is preserved in case of database updates. By default, data is persisted in the following directory on the machine hosting the VM where ESCM is deployed (the Docker host):

`<docker>/data/oscm-db/data:/var/lib/postgresql/data`

Hereafter, the `<docker>` directory is referred to as `/docker`.

Container Communication

The following figure provides an overview of the container communication on a Docker host (a VM):



The host-internal communication always relies on the HTTP protocol, whereas calls from outside of the host are always secured via HTTPS. The platform operator is responsible for opening the indicated ports, which can either be addressed using the FQDN or the floating IP address of the respective container.

To look inside a container:

1. List all containers (even stopped ones) to show the container names:

```
docker ps -a
```

2. Log in to a container using the following command:

```
docker exec -it <container name> /bin/bash
```

For example:

```
docker exec -it oscm-core /bin/bash
```

2 Prequisites and Preparation

For getting started with the setup of ESCM, a Linux system is required with the following software installed:

1. Docker
2. Docker Compose

This system is hereafter referred to as the *Docker host*.

The following system resources are recommended for the Docker host:

1. 2 CPU cores
2. 8 GB of RAM
3. 20 GB of disk space

Note:	This is a minimum configuration for testing purposes only. It is not suitable for production use.

3 Installation

The installation of ESCM consists of the following main steps:

1. If you have received ESCM Docker images in archive files (tar.gz files), importing the Docker images into your local Docker registry.
2. Preparing the installation directory on the Docker host.
3. Preparing configuration files.
4. Preparing Docker Compose files and starting ESCM.

3.1 Importing ESCM Docker Images into a Local Registry

Note: This section is relevant only if you are installing the ESCM Docker images from `tar.gz` files, not directly from DockerHub.

In order to install Docker images from archive files, you need access to a local Docker registry:

Make sure you have command-line access with write permissions to the Docker registry server in your network. If this is not available, you may choose to install such a server in your target environment. Refer to the Docker documentation at *https://docs.docker.com/registry/deploying* for details on how to deploy a Docker registry server.

In the subsequent sections, this registry is referred to as `${REGISTRY}`.

Note: The name of the default DockerHub registry is `servicecatalog`.

Proceed as follows:

1. On your Docker machine, copy the ESCM Docker image archive files into the `/opt` folder.
2. Import the images into your Docker registry using the following Docker command bash script:

```
#!/bin/bash

export REGISTRY=<docker registry service>

docker login ${REGISTRY}

for docker_image in \
        oscm-deployer \
        oscm-core \
        oscm-app \
        oscm-core \
        oscm-birt \
        oscm-db \
        oscm-help \
        oscm-initdb \
; do

       docker load –i /opt/${docker_image}.tar.gz

       docker tag ${REGISTRY}/${docker_image}:v18

       docker push ${REGISTRY}/${docker_image}:v18

done
```

3.2 Preparing the Installation Directory

On the Docker host, you need to create a directory where the various data can be stored, such as persistent database data or configuration files. Hereafter, this directory is referred to as `/docker`.

On the Docker host, execute the following command:

```
mkdir /docker
```

3.3 Preparing Configuration Files

A deployment container is available which prepares configuration file templates where you can specify settings such as mail server, login information, etc.

Use `-v` to mount the `/docker` directory to `/target` in the container:

```
docker run --name deployer1 --rm -v /docker:/
    target ${REGISTRY}/oscm-deployer
```

where `${REGISTRY}` is the name of your local Docker registry or the DockerHub registry.

This command creates two configuration files in the `/docker` directory:

1. `.env`: Configuration settings for Docker, such as images and the base data directory (default: `/docker`)
2. `var.env`: Configuration settings for ESCM and APP, such as mail server, database and other settings.

Edit both files and adjust the configuration settings to your environment. The initial, mandatory settings are stored in the `bss` and `bssapp` databases. They are described in detail in the *Operator's Guide*.

3.4 Preparing Docker Compose Files and Starting ESCM

A second deployment container is available which you can run to do the following:

1. Create the necessary Docker Compose files.
2. Create the necessary subdirectories.
3. Initialize the application databases.
4. Start the application containers.

Execute the following command on your Docker host:

```
docker run --name deployer2 --rm -v /docker:/target
    -v /var/run/docker.sock:/var/run/docker.sock
    -e INITDB=true -e STARTUP=true ${REGISTRY}/oscm-deployer
```

where `${REGISTRY}` is the name of your local Docker registry or the DockerHub registry.

4 Usage

4.1 Accessing the ESCM Administration Portal

After having run the second deployment container, ESCM will take several minutes to start up. The less CPU power you have, the longer it will take. Once everything has started, you may access the ESCM administration portal in your Web browser using the FQDN or IP address you specified earlier in the `var.env` file.

Access the ESCM administration portal in a Web browser using an URL in the following format:

`https://<hostname.fqdn>:<port>/oscm-portal`

`<hostname.fqdn>` is the name and the fully qualified domain name of the machine where ESCM has been deployed. `<port>` is the port to address the machine (default: `8081`), `oscm-portal` is the default context root of ESCM and cannot be changed.

You are prompted for the user ID and password. The initial credentials are as follows:

User ID: `administrator`

Password: `admin123`

It is recommended that you change the initial password in the ESCM administration portal (**Change Password** page in the **Account** menu).

After login, the operator functionality is available in the **Operation** menu.

4.2 Enable Login to APP and Service Controllers

In order to be able to login to the Asynchronous Provisioning Platform (APP) and its service controllers, some settings have to be made in the administration portal:

1. Choose **Manage organization** in the **Operation** menu.
2. Enter `PLATFORM_OPERATOR` in the **Organization ID** field.
3. Enable the following organization roles: **Supplier** and **Technology provider**.
4. Fill in the mandatory fields (red asterisks).
5. Click **Save**.
6. Go to the **Account** menu and choose **Manage users**.
7. Click on `administrator`.
8. Enter your email address.
9. Enable all user roles.
10. Click **Save**.
11. Log out of the administration portal and log in again to enable the changes.
12. Now you can log in to APP and change APP-specific settings:

    `https://<hostname.fqdn>:8881/oscm-app/`

    User name: `administrator`

    Password: `admin123`

13. You can also login to the instance status interface of each service controller:

    `https://<hostname.fqdn>:8881/oscm-app-<controller-name>/`

    `<controller-name>` is either `openstack`, `aws`, `vmware`, `azureARM`, or `shell`.

    User name: `administrator`

Password: `admin123`

4.3 Start Using ESCM

For the initial steps for starting to use ESCM, refer to the *Getting Started* document.

5 Integrating Custom SSL Certificates and Key Files

Certificates are required for ESCM to allow for trusted communication between ESCM and the Asynchronous Provisioning Platform (APP), or an application underlying a technical service . The ESCM deployer has already created a respective directory structure and a suitable Docker Compose configuration. In this way, default certificates have been inserted into the respective containers after deployment, thus communication between ESCM and APP is secured.

It is however possible to use custom SSL keypairs for the application listeners. They may be self-signed or official. Privacy Enhanced Mail (PEM) format is mandatory. This is a container format that may include just the public certificate, or may include an entire certificate chain including public key, private key, and root certificates. It is only necessary to place the respective certificate and/or key files in PEM format into the appropriate directories.

5.1 Importing SSL Key Pairs for the Application Listeners

If you want to use your own SSL key pairs that your application is to use, replace the default key pair by your PEM files in the following directories on your Docker host:

- Private key: `/docker/config/<container-name>/ssl/privkey`
- Public certificate: `/docker/config/<container-name>/ssl/cert`
- Intermediates / chain (optional): `/docker/config/<container-name>/ssl/chain`

Replace `/docker` with the base directory on the Docker host that you use for persisting data, and `<container-name>` with the name of the respective ESCM container, for example, `oscm-core` or `oscm-app`.

The custom certificates must also be placed into the following trusted directory so that a trusted relationship between the containers is established:

- `/docker/config/certs`

The `/docker/config/certs` directory is shared by all containers. By default, if you use your own SSL key pairs, you must also place all the public certificate files here.

For example, if you have a custom SSL keypair for the `oscm-core` container, you need to place the private key into the `/docker/config/oscm-core/ssl/privkey` directory, and the public certificate into the `/docker/config/oscm-core/ssl/cert` directory. Additionally, you need to place the public certificate into the `/docker/config/certs` directory on your Docker host. In this case, a restart of the `oscm-core` and `oscm-app` containers is required.

5.2 Importing Trusted SSL Certificates

If you want your application to trust certain, possibly self-signed SSL certificates, put them in PEM format in the following directory on your Docker host:

`/docker/config/certs`

Replace `/docker` with the base directory on the Docker host that you use for persisting data.

For example, if you want to use the VMware service controller, you need to export the vSphere certificate in PEM format, and copy it to the `/docker/config/certs` directory. Since the VMware service controller is running in the `oscm-app` container, a restart of this container is required.