SAMSUNG ELECTRONICS

Knox E-FOTA On-Premises

Installation and Initial Operation Guide

Version : 1.0

Last Update : Sep 2021

【Document History】

What	Ver.	When
Initial Release	Ver1.0	Sep 2021

Table of Contents

Table of Figures & Tables

【Figures】

【Tables】

PART I: Getting Started

PART 1: Getting Started presents the purpose of this document, the customer infrastructure that is recommended prior to the installation of the Knox E-FOTA On-Premises service, and provides an overview of deliverables to be used during the installation.

1. Introduction

1.1. Purpose of this document

The purpose of this document is to present how to plan for, install, and configure the managed DFM module within the customer's network. This document includes information about how to install and configure 3rd party software, such as Podman, and provides detailed descriptions of the commands used to perform its installations.

This document is intended **for the personnel who are in charge of performing the installation**.

To prepare the installer, this document includes the following tasks:

- Evaluate the customer's network and hardware facilities
- Introduce which modules will be installed to provide this service
- Explain the install flow with DFM Modules
- Explain how to configure the installed DFM Modules with the proper conditions
- Explain how to test if the installed DFM Modules are running as expected

The server infrastructure, hereafter referred to as **DFM Modules** will be installed on the customer's side by Samsung to service the Knox E-FOTA On-Premises environment.

We recommend "The 4-Days Installation" method for this installation, as the customer should understand how they are using this service during this program (see "*Appendix D. A Recommended Schedule for On-Site Installation by CSO/TEO*").

2. Environment Prerequisites

This chapter presents the hardware, software and network facilities required by the DFM. To ensure proper support of E-FOTA On-Premise, the service must be installed upon the following recommended software and hardware infrastructure.

The following recommended items should be prepared by the customer prior to the installation of the Knox E-FOTA On-Premise service by Samsung personnel.

2.1. Recommended Hardware

Table 2-1 below outlines the recommended user environment, including network card for the On-Premise Hardware (H/W) requirements. The customer can choose the correct value depending on the product type—see "***2.3 Recommendation Per each Product Usage***".

Items	Recommended Value	Description
Server CPU Cores	Above 4 or 8 CPU Cores	4 Cores is for PoC Product Above 8 Cores is for Commercial Product
RAM	16 or 32 GB	16GB is for PoC Product 32GB is for Commercial Product
Disk	1TB or 2TB SSD	For DFM Module 1TB (PoC), 2TB (Commercial Product)
	256 GB	For System region (OS and Root filesystem)
Network Card	Above 10 Gbps	

Table 2-1 The recommended hardware for a user work environment to run Knox E-FOTA On-Premises

The recommendations in the above table are the minimum specifications to run the Knox E-FOTA On-Premises service. User performance expectations may require additional infrastructure resources in excess of the minimum specifications.

2.2. Recommended Software

The recommended user work environment, including network, for the On-Premise Software (S/W) requirements are as follows:

Items	Recommended Value	Description
Operating System	Red Hat Enterprise Linux 8.3	
Container Tool	Podman	
MySQL Edition	Enterprise Edition	For Commercial Product

Table 2-2 The recommended hardware for a user work environment to run Knox E-FOTA On-Premises

Refer to "*Appendix C*" for a summary of software recommendations.

2.2.1. Operating System

By default, the DFM Server requires Red Hat Enterprise Linux 8.3 for the OS. It should be installed on 64-bit Intel x86, ARM, or MIPS architectures in order to support Podman.
Selinux mode in the Redhat OS supports permissive and enforcing modes, and the DFM Server can

be installed in each mode.

2.2.2. Podman

Podman is a daemonless container engine for developing, managing, and running OCI Containers on your Linux System. Containers can either be run as root or in rootless mode. Since Redhat Enterprise Linux 8, Podman is officially supported. For Knox E-FOTA On-Premises, Podman is the recommended container tool for Redhat users. Podman installation is described in **Section 4.1.**

2.2.3. Database (MySQL)

The MySQL database contains all service-related data, including device models, their IDs, and policy dependencies in Campaigns.

2.2.4. HTTPS

To use the https protocol between Samsung mobile devices and the DFM Modules, the customer should prepare a DNS hostname (FQDN) and public (or private) SSL certificates.

2.3. Recommendation Per each Product usage

Knox E-FOTA On-Premises has 3 types of product use case architecture recommendations, including 2 Commercial and 1 POC architecture.

2.3.1. Product – "PoC"

The **PoC** product is recommended if a customer wants to use the on-premise service to understand its functions and product configuration clearly prior to purchasing a Commercial Product, along with a small number of devices (clients, until 300 devices). The PoC product can run on lower specification hardware than the Commercial product, but the table below contains the minimum specifications to be running Knox E-FOTA On-Premises. To ensure the service runs as expected, the customer should set up the infrastructure with higher specifications than shown below.

【Minimum Hardware Recommendation】

Items	Recommended Value	Description
Server CPU Cores	4 CPU Cores	
RAM	16 GB	
Disk	1TB SSD	For DFM Module
	256 GB	For System region (OS and Root filesystem)
Network Card	Above 10 Gbps	

Table 2-3 The Minimum Hardware Recommendation for PoC

【Software Recommendation】

Items	Recommended Value	Description
Operating System	Red Hat Enterprise Linux 8.3	
Container Tool	Podman	
MySQL Edition	Community Edition	For continuous Commercial support, recommend Enterprise Edition

Table 2-4 The Software Recommendation for "PoC"

The customer can purchase Redhat OS based on this service package, depending on their service environment. Note that the customer must provide the service infrastructure to the Samsung representative in charge of the installation.

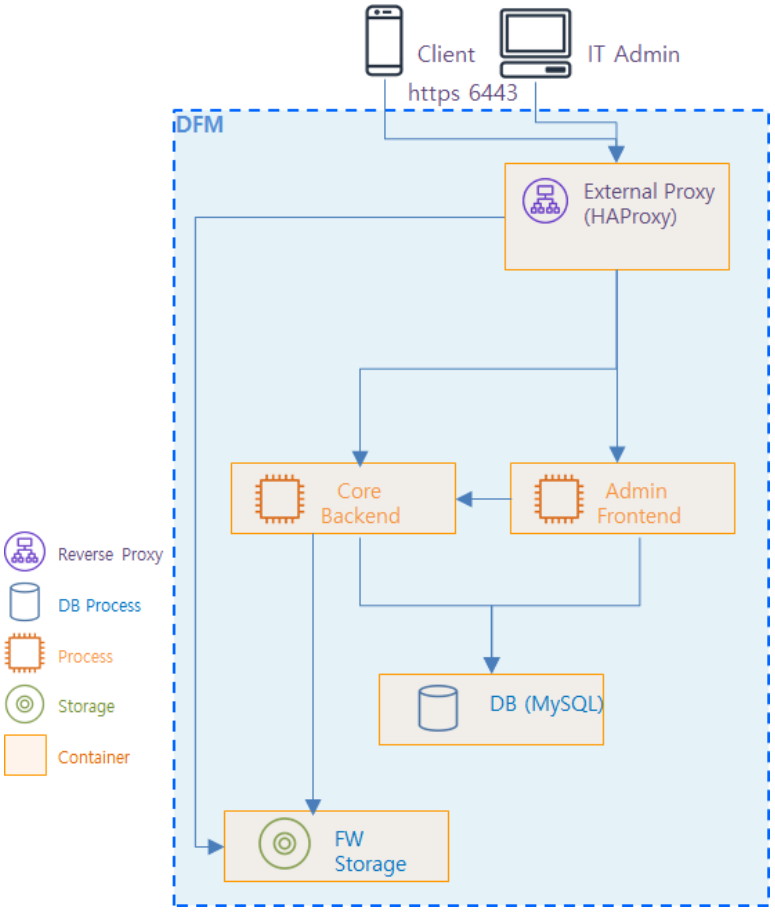

Fig 2-1 Knox E-FOTA On-Premises Product Arch for "PoC"

2.3.2. Product – "Commercial"

The **Commercial** product is recommended for customers who want to use this product with a maximum of 20,000 devices for device firmware updates over-the-air (FOTA), but it also supports more than 20,000 devices.

The recommended specification for the infrastructure is the minimum required to be running the service. To optimize performance expectations, the customer may need to provide infrastructure with higher specifications than the below table to the Samsung representative in charge of the installation.

【Minimum Hardware Recommendation】

Items	Recommended Value	Description
Server CPU Cores	8 CPU Cores	
RAM	32 GB	
Disk	2TB SSD	For DFM Module
	256 GB	For System region (OS and Root filesystem)
Network Card	Above 10 Gbps	

Table 2-5 The Minimum Hardware Recommendation for "Commercial"

【Software Recommendation】

The customer can purchase Redhat OS based on this service package, depending on their service environment. Note that the customer must provide the service infrastructure to the Samsung representative in charge of the installation.

Items	Recommended Value	Description
Operating System	Red Hat Enterprise Linux 8.3	
Container Tool	Podman	
MySQL Edition	Enterprise Edition	

Table 2-6 Software Recommendation for "Commercial"

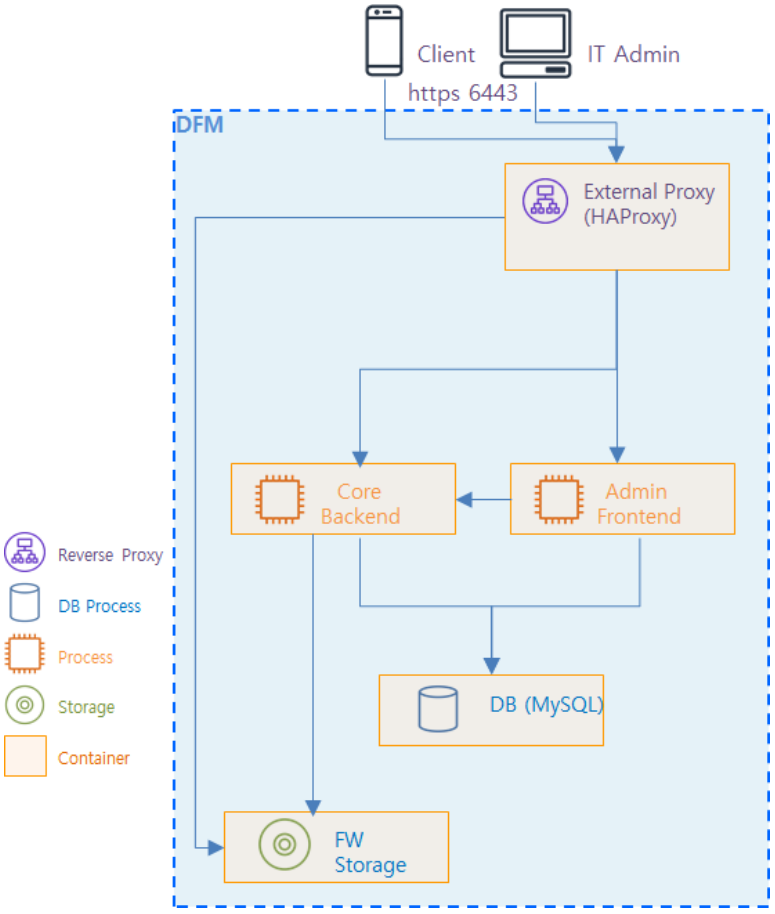

Fig 2-2 Knox E-FOTA On-Premises Product Arch for "Commercial"

3. Deliverables

This chapter describes the actions performed by Samsung to deliver the Knox E-FOTA On-Premises environment.

3.1. DFM Modules

The DFM Module consists of the following core modules:

- **DFM Admin Console Server**: The Frontend module to provide IT admins with an accessible graphical user interface (GUI) on the Google Chrome browser.
- **DFM Core Server**: The Backend module to manage device (client application) actions, integrated into the device using RESTful APIs from the client.
- **DFM Database**: The MySQL-based database contains all service-related data, including device models, their IDs, and policy dependencies in Campaigns.
- **DFM Firmware Storage Management**: The firmware files for downloaded files from the client application.
- **Proxy:** Used for redirection between outer and DFM modules, and for AP Gateway and TLS/SSL termination.

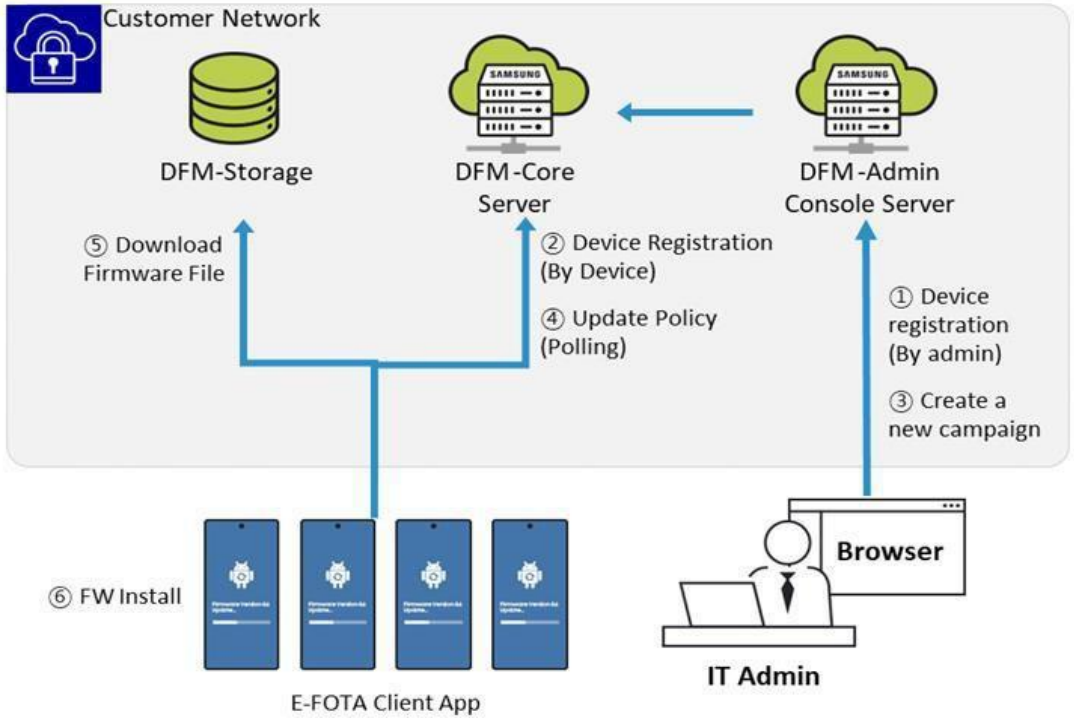

Fig 3-1 **Knox E-FOTA On-Premises Conceptual Architecture**

3.2. Security Considerations

To improve the default security of the Samsung deliverable, it must be implemented using the following standards.

3.2.1. HTTPS and Network encryption

The DFM Module uses HTTPS TLS-based encryption to enhance the security of transactions. The Transport Layer Security (TLS) protocol provides data encryption and verification between applications and servers in scenarios where data is being sent across an insecure network—for example, when working with the DFM Module.

HTTPS header fields are components of the header section of HTTPS request and response messages. They define the operating parameters of a HTTPS transaction. The load balancer and reverse proxy are in front of the DFM Module queries.

3.3. Supported Browser

PLEASE NOTE that this version of the DFM Console UI is designed for **Google Chrome** only.

PART II: Installation and Validation

PART II: Installation and Validation describes how to install the Knox E-FOTA On-Premises service on the customer-provided infrastructure, and how to validate the installed service infrastructure.

4. Installation & Configuration

This chapter explains the first-time installation flow with proper configuration conditions of the DFM Modules. Steps in this chapter run only once during initial installation.
For **Redhat OS** setups, the installation method varies based on the user privilege (root or non-root) and the Selinux mode (permissive or enforcing).

Refer to the table below to determine the installation method to use. For example, if you should install by rootless-enforcing, proceed to the **【CASE Red Hat 3】** installation method.

Items		User privilege		Description
		root	rootless	
Selinux mode	Permissive	CASE Red Hat 1	CASE Red Hat3	
	enforcing	CASE Red Hat 2		

Table 4-1 The Red hat Case

4.1. (Prerequisites) Install Package

【CASE RedHat 1 , CASE RedHat 2】
sudo yum install -y podman podman-docker podman-plugins dnsmasq udica
【CASE Red Hat 3】
sudo yum install -y podman udica

4.2. (STEP01) Create Service Account and Login

The DFM Module is logged in with **a dedicated service account** and operates with the privileges of the account. Therefore, the dedicated service account must be created in the server.
We recommend you create a service account before you start the installation.
DFM module supports both root and rootless (non-root) users.

The section below shows how to create a rootless user (**【CASE Red Hat 2】**) and connect:

```
sudo useradd -c “{your-name}” {your-user}

Example)
 sudo useradd -c “nightwatch” nightwatch
```

To connect using a created user:

ssh {your-user}@localhost [-p {port}]

Example)
ssh nightwatch@localhost

or if you use port 9000
ssh nightwatch@localhost -p 9000

4.3. (STEP02) Prepare “Disk partition & mount” for DFM modules

The DFM module is installed and operates in the below directory on the **dedicated disk**. Check if the dedicated disk exists and the “partition & mount” is ready, in case the customer has not worked with the disk partition for the DMF module before.

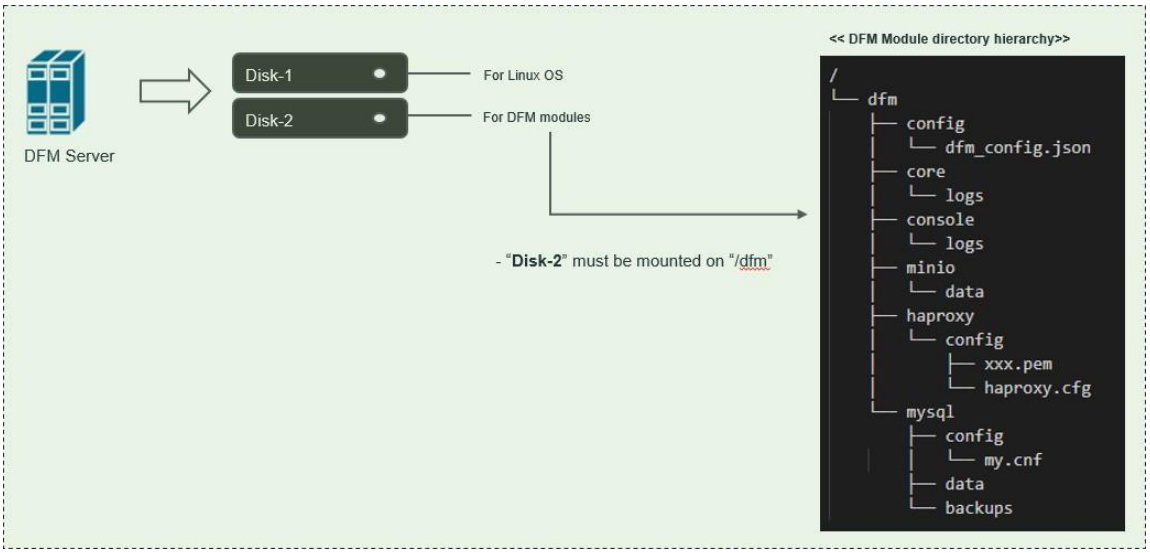

Fig 4-1 An Disk Partitions for DMF Module

For example, we assume that two disks (“sda” and “sdb”) exist.

【CASE01】Disk is Ready

If the disks exist, we don’t need to format and mount them. Now, let’s check the disk information:

sudo lsblk -p

```
NAME         MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
/dev/sda     202:0    0    1T  0 disk
└─/dev/sda1  202:1    0    1T  0 part /
/dev/sdb     202:80   0    1T  0 disk
```

sudo lsblk -f

```
NAME    FSTYPE   LABEL          UUID                                 MOUNTPOINT
sda
└─sda1  ext4     xxxxxxxx-rootfs 6156ec80-9446-4eb1-95e0-9ae6b7a46187 /
sdb     ext4                     d3269ceb-4418-45d0-ba68-d6b906e0595d /dfm
```

⇨ “sdb” is already formatted and mounted on **/dfm**

sudo file -s /dev/sdb

```
/dev/sdb: Linux rev 1.0 ext4 filesystem data, UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d (extents) (64bit) (large files) (huge files)
```

【CASE02】Disk is NOT Ready: it is not formatted

If the disk is not ready, it needs to be formatted and mounted on **/dfm**.

Now, let’s check the disk information:

sudo lsblk -p

```
NAME         MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
/dev/sda     202:0    0    1T  0 disk
└─/dev/sda1  202:1    0    1T  0 part /
/dev/sdb     202:80   0    1T  0 disk
```

sudo lsblk -f

```
NAME    FSTYPE   LABEL          UUID                                 MOUNTPOINT
sda
└─sda1  ext4     xxxxxxxx-rootfs 6156ec80-9446-4eb1-95e0-9ae6b7a46187 /
sdb
```

⇨ “sdb” is NOT formatted

sudo file -s /dev/sdb

```
/dev/sdb: data
```

⇨ This means that the disk needs to be formatted

1) Format with ext4 file-system

```
sudo file -s /dev/sdb
```

```
sudo mkfs -t ext4 /dev/sdb
mke2fs 1.44.1 (24-Mar-2018)
Creating filesystem with 2621440 4k blocks and 655360 inodes
Filesystem UUID: d3269ceb-4418-45d0-ba68-d6b906e0595d
Superblock backups stored on blocks:
    32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Allocating group tables: done
Writing inode tables: done
Creating journal (16384 blocks): done
Writing superblocks and filesystem accounting information: done
```

2) Check if the disk is formatted

```
sudo mkfs -t ext4 /dev/sdb
```

```
/dev/sdb: Linux rev 1.0 ext4 filesystem data, UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d (extents) (64bit) (large files) (huge files)
```

3) Mount "/dev/sdb" on /dfm

```
// create directory to mount
sudo mkdir /dfm

// mount
sudo mount /dev/sdb /dfm
```

4) Verify

```
df -h
```

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb        9.8G   37M  9.3G   1% /dfm
```

【CASE03】Disk is NOT Ready : it is already formatted but not yet mounted on /dfm

If the disk is formatted but not yet mounted, it needs to be mounted on **/dfm**. Now, let's check the disk information:

```
sudo lsblk -p
```

```
NAME         MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
/dev/sda     202:0    0    1T  0 disk
└─/dev/sda1 202:1    0    1T  0 part /
/dev/sdb     202:80   0    1T  0 disk
```

```
sudo lsblk -f
```

```
NAME    FSTYPE   LABEL          UUID                                   MOUNTPOINT
sda
└─sda1  ext4     xxxxxxxx-rootfs 6156ec80-9446-4eb1-95e0-9ae6b7a46187 /
sdb     ext4                    d3269ceb-4418-45d0-ba68-d6b906e0595d
```

⇨ “sdb” ” is formatted but not yet mounted

1) Mount /dev/sdb on /dfm

```
// create directory to mount
sudo mkdir /dfm

// mount
sudo mount /dev/sdb /dfm
```

2) Verify

```
df -h
```

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb        9.8G   37M  9.3G   1% /dfm
```

*) We recommend that the **customer's IT manager** sets the boot script so that **the dedicated disk** is auto-mounted when the server is booted.

4.4. (STEP03) Create Service Directories

A separated service directory configuration is required to install and operate the Samsung DFM Module. The service account must have “**read / write / execute**” permissions to the service directory. The service directory should be mounted in a different device location from the OS installation area.

【Service Directory List】

/dfm/mysql/config
- that is where the config file is referenced when the mysql server starts.

/dfm/mysql/data
- that is where databases are created when mysql server runs.

/dfm/mysql/backups
- that is where databases are backed-up when mysql server runs.

/dfm/minio/data/dfm-agent-storage
- that is where efota client APK files are uploaded when minio server runs.

/dfm/minio/data/dfm-fw-storage
- that is where firmware binary files are uploaded when minio server runs.

/dfm/haproxy/config
- that is where the config file is referenced when haproxy server starts.

/ dfm/console/logs
- that is where log files are generated when admin console server runs.

/dfm/core/logs
- that is where log files are generated when core server runs.

/dfm/config
- that is where the config file contains the information needed to run the DFM module.

【CASE Red Hat 3】

/dfm/bin
- that is where dfm cli file is referenced when dfm module start, terminate or create
- we need to create an additional directory in【CASE Red Hat 3】.

Now, let's create each service directory.

In【**CASE Red Hat 3**】, we need to create an additional directory.

```
sudo mkdir -p /dfm/mysql/config
sudo mkdir -p /dfm/mysql/data
sudo mkdir -p /dfm/mysql/backups
sudo mkdir -p /dfm/minio/data/dfm-agent-storage
sudo mkdir -p /dfm/minio/data/dfm-fw-storage
sudo mkdir -p /dfm/haproxy/config
sudo mkdir -p /dfm/console/logs
sudo mkdir -p /dfm/core/logs
sudo mkdir -p /dfm/config

【CASE Red Hat 3】
sudo mkdir -p /dfm/bin
```

Set the service account's permission for the created service directory.

We assume that you are using the "**nightwatch**" account.

```
sudo chown -R nightwatch:nightwatch /dfm
sudo chown -R nightwatch:nightwatch /dfm/mysql
sudo chown -R nightwatch:nightwatch /dfm/minio
sudo chown -R nightwatch:nightwatch /dfm/haproxy
sudo chown -R nightwatch:nightwatch /dfm/mysql/config
sudo chown -R nightwatch:nightwatch /dfm/mysql/data
sudo chown -R nightwatch:nightwatch /dfm/mysql/backups
sudo chown -R nightwatch:nightwatch /dfm/minio/data
sudo chown -R nightwatch:nightwatch /dfm/minio/data/dfm-agent-storage
sudo chown -R nightwatch:nightwatch /dfm/minio/data/dfm-fw-storage
sudo chown -R nightwatch:nightwatch /dfm/haproxy/config
sudo chown -R nightwatch:nightwatch /dfm/console/logs
sudo chown -R nightwatch:nightwatch /dfm/core/logs
sudo chown -R nightwatch:nightwatch /dfm/config

【CASE Red Hat 3】
sudo chown –R nightwatch:nightwatch /dfm/bin
```

4.4.1. Authorizing the Mysql data folder

To write a file from the Mysql container to the MySql folder of the host server, run the following command.

The section below refers to 【**CASE Red Hat 2, CASE Red Hat 3**】in Redhat.

We assume that you are using the "**nightwatch**" account.

```
podman unshare chown -R nightwatch:nightwatch /dfm/mysql/data
```

4.5. (STEP04) Install DFM Module Package

The DFM Module is delivered as a tar compress file.
Packages used for each case are different.

Items		User privilige		Description
		root	rootless	
Selinux mode	Permissive	CASE Red Hat 1 **sec-dfm_{version}.tar.gz**	CASE Red Hat3 **sec-dfm_{version}-rootless.tar.gz**	
	enforcing	CASE Red Hat 2 **sec-dfm_{version}-root-enforcing.tar.gz**		

Table 4-2 **The Red hat DFM Module package Case**

For example, if you install as a rootless user, proceed using the **"sec-dfm_{version}-rootless.tar.gz"** file.

This package contains the following resources:

- executable binary (dfm): managed command to run DFM module
- images: podman image about DFM module
- sql query file: DFM module's DB data to initialize mysql
- mysql config file (my.cnf): config file for mysql
- haproxy config file (haproxy.cfg): config file for haproxy
- dfm config file (dfm_config.json): config file for DFM module
- semodule file: config file for selinux

To extract these resources, the host must unpack the files within the following locations:

- executable binary (dfm):
    * /tmp/dfm/bin/dfm

- images:
    * /tmp/dfm/images/dfm-haproxy-xxx.tar
    * /tmp/dfm/images/dfm-minio-xxx.tar
    * /tmp/dfm/images/dfm-mysql-xxx.tar
    * /tmp/dfm/images/dfm-console-xxx.tar
    * /tmp/dfm/images/dfm-core-xxx.tar
-
- sql query file
    * /tmp/dfm/mysql-query/init_db.sql
    * /tmp/dfm/mysql-query/init_dfm_core.sql
    * /tmp/dfm/mysql-query/init_dfm_console.sql
-
- mysql config file
    * /tmp/dfm/mysql-config/my.cnf
-
- haproxy config
    * /tmp/dfm/haproxy-config/haproxy.cfg
-
- dfm config file
    * /tmp/dfm/dfm_config.json
-
- semodule file
    * /tmp/dfm/selinux_cli/dfm_console_t.cil
    * /tmp/dfm/selinux_cli/dfm_core_t.cil
    * /tmp/dfm/selinux_cli/dfm_mysql_t.cil
    * /tmp/dfm/selinux_cli/dfm_proxy_t.cil
    * /tmp/dfm/selinux_cli/dfm_minio_t.cil

The following is a command showing how to extract the package:

```
tar -zxvf sec-dfm_xxx-{package type}.tar.gz

example)
tar -zxvf  sec-dfm_1.0.1.2-rootless.tar.gz
sec-dfm_1.0.1.2-rootless/
sec-dfm_1.0.1.2-rootless/tmp/
sec-dfm_1.0.1.2-rootless/tmp/dfm/
sec-dfm_1.0.1.2-rootless/tmp/dfm/mysql-query/
sec-dfm_1.0.1.2-rootless/tmp/dfm/mysql-query/init_dfm_console.sql
sec-dfm_1.0.1.2-rootless/tmp/dfm/mysql-query/init_dfm_core.sql
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/mysql-enterprise-server-8.0.20.tar
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/haproxy-debian-2.1.4.tar
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/dfm-console_1.0.1.2-rootless.tar
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/dfm-core:1.0.1.2-rootless.tar
sec-dfm_1.0.1.2-rootless/tmp/dfm/images/minio-RELEASE.2020-06-01T17-28-03Z.tar
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/dfm_minio_t.cil
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/dfm_proxy_t.cil
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/dfm_core_t.cil
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/dfm_console_t.cil
sec-dfm_1.0.1.2-rootless/tmp/dfm/selinux-cil/dfm_mysql_t.cil
sec-dfm_1.0.1.2-rootless/usr/
sec-dfm_1.0.1.2-rootless/usr/bin/
sec-dfm_1.0.1.2-rootless/usr/bin/dfm
```

Next, check if the necessary files exist:

1) check dfm file
ls /usr/bin/dfm
/usr/bin/dfm

2) check docker images
ls /tmp/dfm/images/ -l
total 810548
-rw-rw-r-- 1 dfm-console_1.0.1.2-rootless.tar
-rw-rw-r-- 1 dfm-core_1.0.1.2-rootless.tar
-rw-rw-r-- 1 haproxy-debian-2.1.4.tar
-rw-rw-r-- 1 dfm-minio-RELEASE.2020-01-25T02-50-51Z.health.tar
-rw-rw-r-- 1 dfm-mysql-5.7.25.health.tar

3) check sql query file
ls /tmp/dfm/mysql-query/ -l
total 2076
-rw-r--r-- 1 init_db.sql
-rw-r--r-- 1 init_dfm_console.sql
-rw-r--r-- 1 init_dfm_core.sql

4) check mysql config file : my.cnf
ls /tmp/dfm/mysql-config/ -l
total 4
-rw-r--r-- my.cnf

5) haproxy config file : haproxy.cfg
ls /tmp/dfm/haproxy-config/
haproxy.cfg

6) dfm config file : dfm_config.json
ls /tmp/dfm/dfm_config.json
/tmp/dfm/dfm_config.json
7) semodule file:
ls /tmp/dfm/selinux_cli/
total 20
-rw-rw-r-- 1 dfm_console_t.cil
-rw-rw-r-- 1 dfm_core_t.cil
-rw-rw-r-- 1 dfm_proxy_t.cil
-rw-rw-r-- 1 dfm_minio_t.cil
-rw-rw-r-- 1 dfm_mysql_t.cil

4.6. Register SElinux Policy

This section is intended for users using the rootless or enforcing modes of Redhat.
Only the **【CASE Red Hat 2, CASE Red Hat 3】** users described in **4.** should execute the following commands:

```
sudo semodule -i *    /tmp/dfm/selinux_cli/dfm_console_t.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,net_container.cil}

sudo semodule -i *    /tmp/dfm/selinux_cli/dfm_core_t.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,net_container.cil}

sudo semodule -i *    /tmp/dfm/selinux_cli/dfm_mysql_t.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,net_container.cil}

sudo semodule -i *    /tmp/dfm/selinux_cli/dfm_minio_t.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,net_container.cil}

sudo semodule -i *    /tmp/dfm/selinux_cli/dfm_proxy_t.cil
/usr/share/udica/templates/{base_container.cil,net_container.cil,net_container.cil}
```

Check if the Selinux policy is registered:

```
sudo semodule -l | grep dfm_
```

4.7. (STEP05) Load Podman Image

Next, register the Podman Images that were unpacked at "**/tmp/dfm/images**". The loaded Podman Images are used when the container is driven. The following shows how to load each Podman Image using Podman commands:

```
podman load -i /tmp/dfm/images/dfm-haproxy-xxx.tar
podman load -i /tmp/dfm/images/dfm-minio-xxx.tar
podman load -i /tmp/dfm/images/dfm-mysql-xxx.tar
podman load -i /tmp/dfm/images/dfm-console-xxx.tar
podman load -i /tmp/dfm/images/dfm-core-xxx.tar
```

Next, check if the 5 Podman images were loaded. Use the "podman images" command:

Example)

podman images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
localhost/dfm-console	1.0.1.2-rootless	62c236d15854	About an hour ago	191MB
localhost/dfm-core	1.0.1.2-rootless	d63dec253531	About an hour ago	149MB
localhost/minio/minio	RELEASE.2020-06-01T17-28-03Z	88bf690bd83f	6 days ago	57.3MB
localhost/haproxytech/haproxy-debian	2.1.4	1cc730cc6555	8 days ago	123MB
localhost/mysql/enterprise-server	8.0	aee72e79b55b	8 days ago	472MB

4.8. (STEP06) Copy Configuration files

After loading the Podman images, copy the following configuration files into the service directory from the unpacked resources directory.

We assume that you are using the "**nightwatch**" account.

- copy executable binary

【CASE Red Hat 1, CASE Red Hat 2】

```
// copy executable binary
  cp /tmp/dfm/bin/dfm /usr/bin/

// Set executable
  chmod 755 /usr/bin/dfm

// create empty file(podman-docker package용)
  touch /etc/containers/nodocker
```

【CASE Red Hat 3】

```
// copy executable binary
  cp /tmp/dfm/bin/dfm /dfm/bin/

// Set executable
  chmod 755 /dfm/bin/dfm

// set executable path(optional)
// modify bashrc file
  vi ~/.bashrc
// add PATH
  export PATH=$PATH:/dfm/bin

// apply modified bashrc file
  source ~/.bashrc
```

- **copy mysql config file**:

```
// copy configuration file
  cp /tmp/dfm/mysql-config/my.cnf /dfm/mysql/config

// Set the service account's permission to the configuration file.
  sudo chown -R nightwatch:nightwatch /dfm/mysql/config
```

- **copy haproxy config file**:

```
// copy configuration file
  cp  /tmp/dfm/haproxy-config/haproxy.cfg  /dfm/haproxy/config

// copy error files
  cp -rf /tmp/dfm/haproxy-config/errors/ /dfm/haproxy/config

 //Set the service account's permission to the configuration file.
    sudo chown -R nightwatch:nightwatch/dfm/haproxy/config
```

- **copy dfm config file**:

```
// copy configuration file
  cp /tmp/dfm/dfm_config.json /dfm/config

//Set the service account's permission to the configuration file.
  sudo chown -R nightwatch:nightwatch /dfm/config
```

4.9. (STEP07) Set-up Configuration

In this step, we will set up the initial configuration information needed for the DFM module to run as a Container.

【Configuration List】

- host_ip: Static IP for DFM server.
- listen_port: External listen port at server for DFM module to be accessed.
- listen_scheme: url scheme(http or https) for DFM module to be accessed.
- access_address: domain-based or ip-based
- access_scheme: http or https
- access_port: public port
- public_endpoint: {access_scheme}://{access_address}:{access_port}

In order to properly configure this service after installation, check the customer's network environment in advance. Be sure to check and verify any port-forwarding mapping (NAT) in the customer's network.

Here are a few sample use-cases:

【Use Case 1】IP-based Access Environment

This environment reflects a real-world network environment. The host IP address is not the same as the public IP address and the CP port number between the public network side and the customer internal network side (including DFM Modules) may be different.

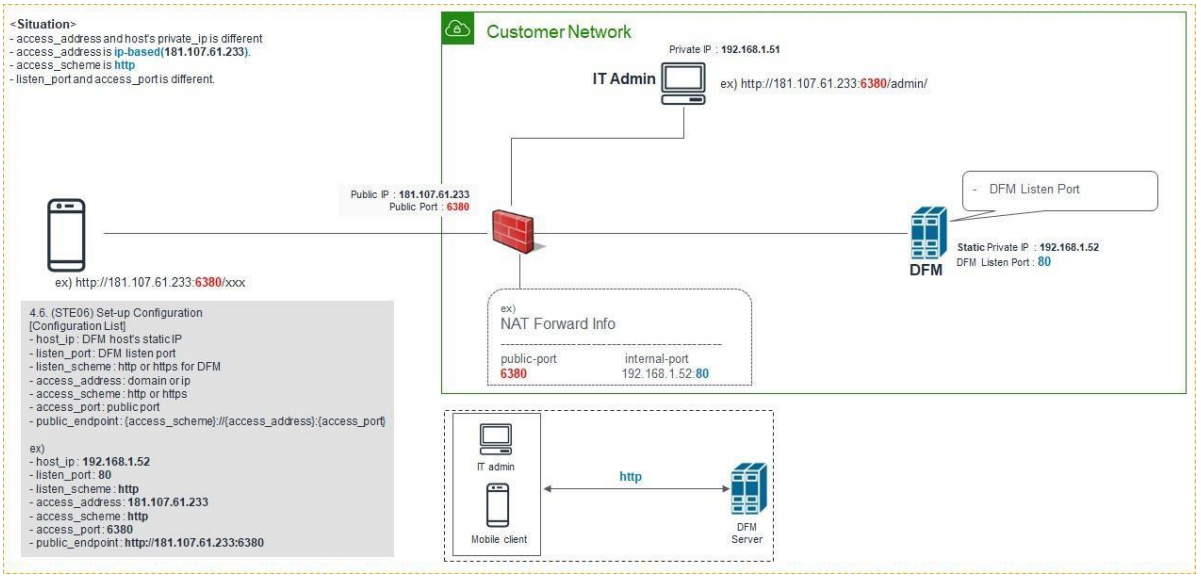

Fig 4-2 **IP-based Access Environment**

【Use Case 2】Domain-based Access Environment

This environment represents a Domain name-based network environment. You can check the network using the Domain name instead of the IP address.

2-1. (Type A) Using **HTTP**

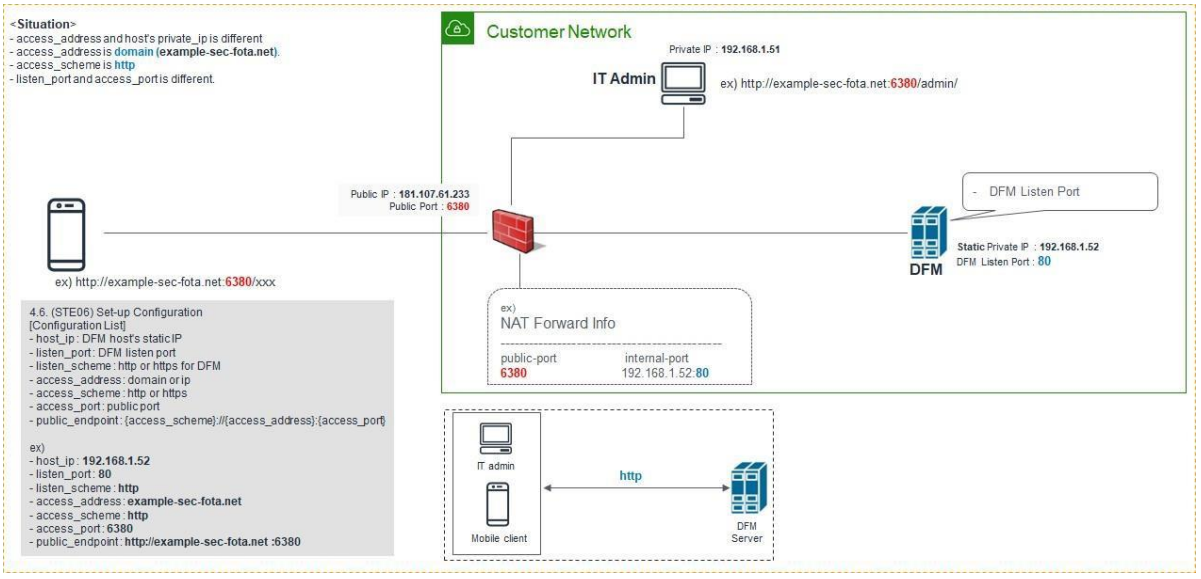

Fig 4-3 **Domain-Based Access Environment (Type A)**

2-2. (Type B) Using **HTTPS** - Customer's LB processes TLS/SSL Termination)

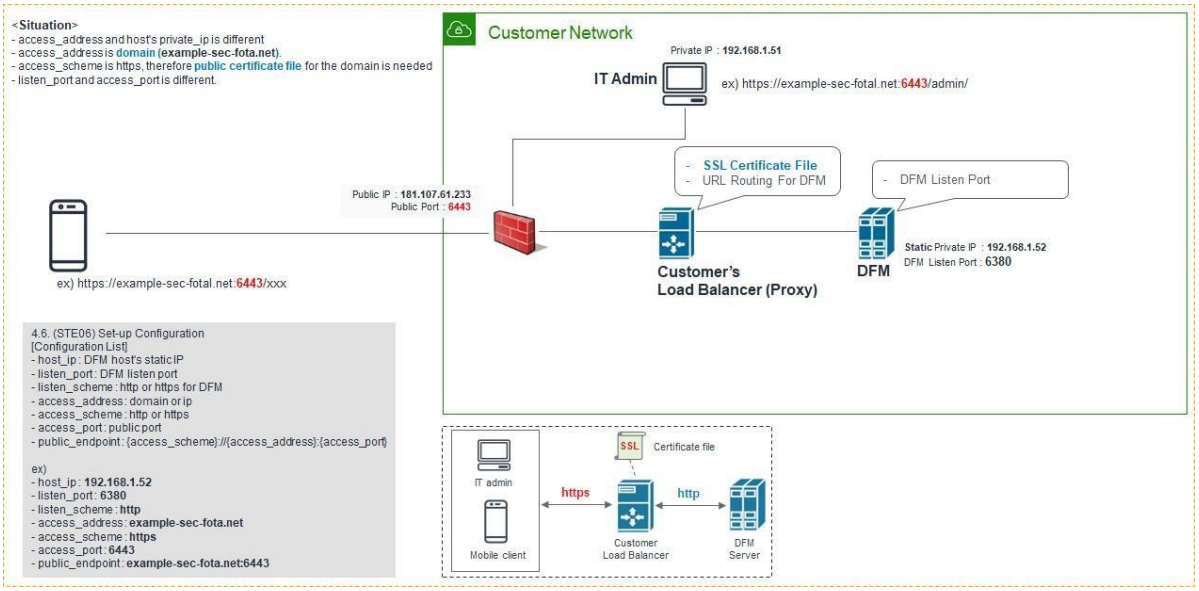

Fig 4-4 **Domain-Based Access Environment (Type B)**

2-3. (Type C) Using **HTTPS** - DFM processes TLS/SSL Termination

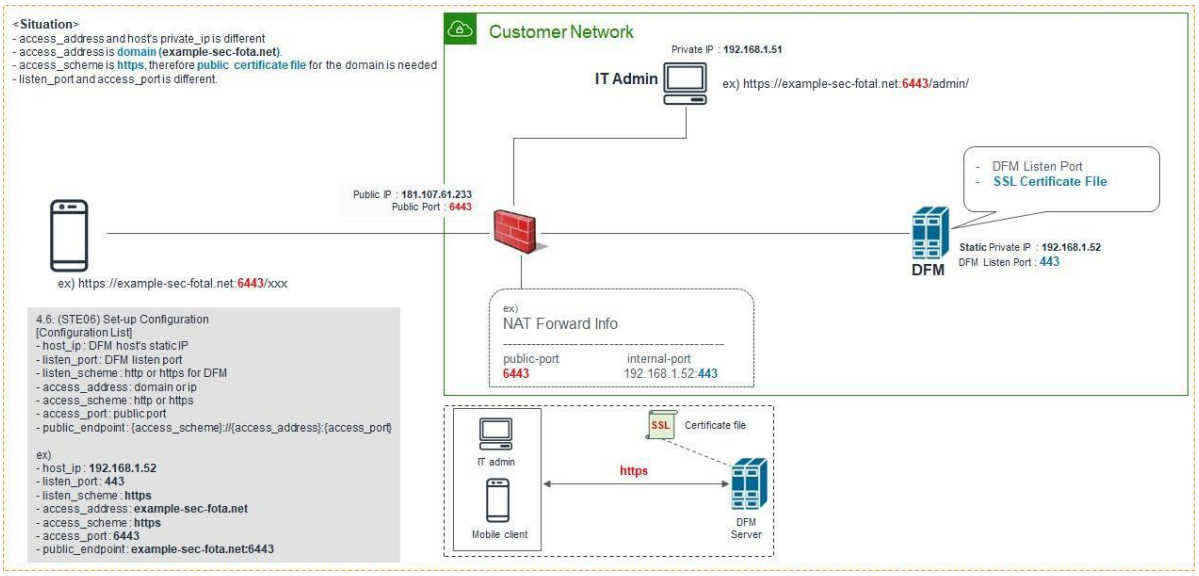

Fig 4-5 **Domain-Based Access Environment (Type C)**

The following is **an example** of how to execute the command to set the above configurations:

2.3.1 (CASE01) IP-Based

- host_ip: 192.168.1.52
- listen_port: 80
- listen_scheme: http
- access_address: 181.107.61.233
- access_scheme: http
- access_port: 6380

The following shows the commands:

```
dfm config set host_ip="192.168.1.52"
dfm config set listen_port="80"
dfm config set listen_scheme="http"
dfm config set access_address="181.107.61.233"
dfm config set access_scheme="http"
dfm config set access_port="6380"
```

Next, check if the configured value is correct. Use the "**dfm config get {*key*}**" command:

```
Example)
dfm config get host_ip
 192.168.1.52

 dfm config get listen_port
 80

 dfm config get listen_scheme
 http

 dfm config get access_address
 181.107.61.233

 dfm config get access_scheme
 http

 dfm config get access_port
 6380
```

2.3.2 (CASE02) Domain-Based

⇦ (**Type ①**) Using **HTTP**

```
- host_ip: 192.168.1.52
- listen_port: 80
- listen_scheme: http
- access_address:  example-sec-fota.net
- access_scheme: http
- access_port: 6380
```

The following shows the commands:

```
dfm config set host_ip="192.168.1.52"
dfm config set listen_port="80"
dfm config set listen_scheme="http"
dfm config set access_address="example-sec-fota.net"
dfm config set access_scheme="http"
dfm config set access_port="6380"
```

Next, check if the configured value is correct. Use the "**dfm config get {*key*}**" command:

Example)

dfm config get host_ip
192.168.1.52

dfm config get listen_port
80

dfm config get listen_scheme
http

dfm config get access_address
example-sec-fota.net

dfm config get access_scheme
http

dfm config get access_port
6380

⇦ (**Type ②**) Using **HTTPS** - Customer's LB processes TLS/SSL Termination

- host_ip: 192.168.1.52
- listen_port: 6380
- listen_scheme: http
- access_address: example-sec-fota.net
- access_scheme: https
- access_port: 6443

The following shows the commands:

```
dfm config set host_ip="192.168.1.52"
dfm config set listen_port="6380"
dfm config set listen_scheme="http"
dfm config set access_address="example-sec-fota.net"
dfm config set access_scheme="https"
dfm config set access_port="6443"
```

Next, check if the configured value is correct. Use the "**dfm config get {*key*}**" command:

Example)
dfm config get host_ip
192.168.1.52

dfm config get listen_port
6380

dfm config get listen_scheme
http

dfm config get access_address

example-sec-fota.net

dfm config get access_scheme
https

dfm config get access_port
6443

⇦ (**Type ③**) Using **HTTPS** - DFM processes TLS/SSL Termination

- host_ip: 192.168.1.52
- listen_port: 443
- listen_scheme: https
- access_address: example-sec-fota.net
- access_scheme: https
- access_port: 6443

The following shows the commands:

dfm config set host_ip="192.168.1.52"
dfm config set listen_port="443"
dfm config set listen_scheme="https"
dfm config set access_address="example-sec-fota.net"
dfm config set access_scheme="https"
dfm config set access_port="6443"

Next, check if the configured value is correct. Use the "**dfm config get {*key*}**" command:

Example)
dfm config get host_ip
192.168.1.52

dfm config get listen_port
443

dfm config get listen_scheme
https

dfm config get access_address
example-sec-fota.net

dfm config get access_scheme
https

dfm config get access_port
6443

4.9.1. Configure Access port

In Redhat OS, Connection is restricted for ports below 1024 for【**CASE Red Hat 2, CASE Red Hat 3**】.
We need to set it to listen_port, which was set above:

sudo sysctl net.ipv4.ip_unprivileged_port_start={port number}

Example)

// check listen_port
dfm config get listen_port
443
// open port to listen_port
sudo sysctl net.ipv4.ip_unprivileged_port_start=443

4.10. (STEP08) Configure HAProxy

If the external connection type is "**https**", the customer must prepare **1)** the access domain they were issued, **2)** a public certificate for the domain in advance. If the customer is using IP address-based addressing rather than DNS, **this step may be skipped**.

If "ingress_url_scheme" is set to "**https**" on the "***4.7. (STEP07) Set-up Configuration***", this step must be completed.

I. HTTPS Handling

There are two possibilities for TLS/SSL Termination:

1) On Customer's Load Balancer (Proxy)

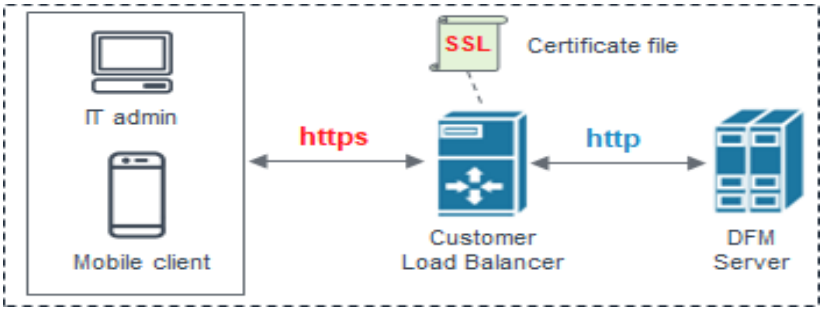

Fig 4-6 **On Customer's Load Balancer (Proxy)**

In this case, the customer's IT manager will operate "public certificate" on its own Load Balancer.

Be careful to comment the "**bind *:443 …**" line and uncomment "**#http-response replace- value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set**" line in the haproxy.cfg file:

vi /dfm/haproxy/config/haproxy.cfg

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
frontend fe_web
    bind *:80
    #bind *:443 ssl crt /usr/local/etc/haproxy/example-sec-fota.net.pem
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
backend dfmConsoleBackend
        mode http
        acl logout_path_set var(txn.path) path /admin/logout
        http-request set-header X-Forwarded-Port %[dst_port]
        http-request add-header X-Forwarded-Proto https if { ssl_fc }

        option httpchk GET /admin/health/live
        http-check expect status 200
        default-server inter 5s fall 3 rise 2

        # if DFM Server is behind customer's Load-Balancer and also customer's Load-Balancer provides ssl termination.
        http-response replace-value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set
        # otherwise
        #http-response replace-value Location (.*) %[var(txn.scheme)]://%[var(txn.host)]/admin/ if logout_path_set

        server dfm-console dfm-console:10050 check resolvers docker init-addr libc,none
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

Since the DFM server can no longer add “Location Header” in response, the **Customer’s Load Balancer must provide the corresponding function**. If the Load Balancer does not provide this function, the user cannot log out after logging into the “admin console webpage” on the DFM.

2) On DFM Server

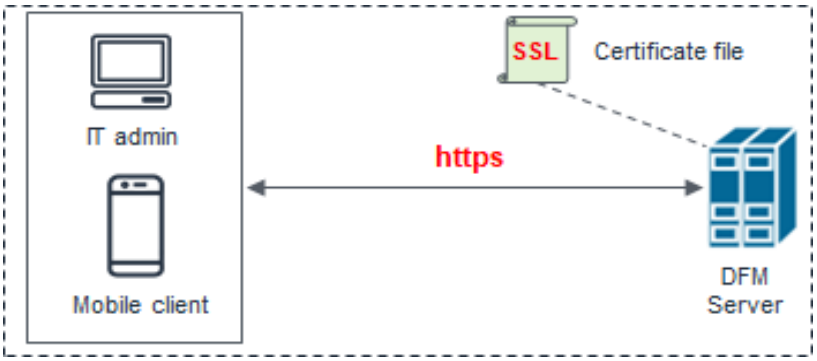

Fig 4-7 On DFM Server

In this case, we need to configure TLS/SSL on our DFM Server. Follow the below steps to do so.

The following assumes that the "**example-sec-fota.net.pem**" file is the public certificate issued by the customer. The public certificate must be copied into haproxy's config folder, and the "haproxy.cfg" file must be edited to change the bind port information and certificate configuration.

- The **crt** parameter identifies the location of the **PEM-formatted** SSL certificate

- **This certificate file** should contain **both the public certificate and private key**
- How to generate the unified certificate for the issued certificate file:

  For example: we assume that you have the below 4 files and the domain's name is **onptest.samsung-efota-test**

  - cert.pem
  - chain.pem
  - fullchain.pem: cert.pem and chain.pem combined
  - privkey.pem

  ⇨ sudo -E bash -c 'cat fullchain.pem privkey.pem > onptest.samsung-efota-test.pem'

  ⇨ '**onptest.samsung-efota-test.pem**' is the unified certificate file

We assume that you are using the "**nightwatch**" account.

Be careful to uncomment the "**bind *:443 …**" line and uncomment the "**#http-response replace-value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set**" line in the haproxy.cfg file:

```
cp example-sec-fota.net.pem /dfm/haproxy/config
sudo chown nightwatch:nightwatch /dfm/haproxy/config/example-sec-fota.net.pem
sudo chmod 600 /dfm/haproxy/config/example-sec-fota.net.pem
vi /dfm/haproxy/config/haproxy.cfg

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
frontend fe_web
    bind *:80
    bind *:443 ssl crt /usr/local/etc/haproxy/example-sec-fota.net.pem
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
backend dfmConsoleBackend
        mode http
        acl logout_path_set var(txn.path) path /admin/logout
        http-request set-header X-Forwarded-Port %[dst_port]
        http-request add-header X-Forwarded-Proto https if { ssl_fc }

        option httpchk GET /admin/health/live
        http-check expect status 200
        default-server inter 5s fall 3 rise 2

        # if DFM Server is behind customer's Load-Balancer and also customer's Load-Balancer provides ssl termination.
        #http-response replace-value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set
        # otherwise
        http-response replace-value Location (.*) %[var(txn.scheme)]://%[var(txn.host)]/admin/ if logout_path_set

        server dfm-console dfm-console:10050 check resolvers docker init-addr libc,none
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

II. HTTP Handling

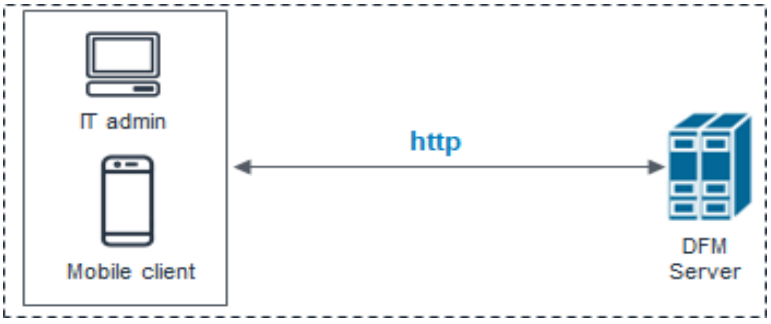

Fig 4-8 On DFM Server

Be careful to comment out the "**bind *:443 …**" line in the haproxy.cfg file in a HTTP-only (non HTTPS) configuration.

【Example】

vi /dfm/haproxy/config/haproxy.cfg

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
frontend fe_web
    bind *:80
    #bind *:443 ssl crt /usr/local/etc/haproxy/example-sec-fota.net.pem
```

```
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
backend dfmConsoleBackend
        mode http
        acl logout_path_set var(txn.path) path /admin/logout
        http-request set-header X-Forwarded-Port %[dst_port]
        http-request add-header X-Forwarded-Proto https if { ssl_fc }

        option httpchk GET /admin/health/live
        http-check expect status 200
        default-server inter 5s fall 3 rise 2

        # if DFM Server is behind customer's Load-Balancer and also customer's Load-Balancer provides ssl termination.
        #http-response replace-value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set
        # otherwise
        http-response replace-value Location (.*) %[var(txn.scheme)]://%[var(txn.host)]/admin/ if logout_path_set

        server dfm-console dfm-console:10050 check resolvers docker init-addr libc,none
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

4.11. (STEP09) Create Container Network

The DFM Module is a process executed on a container basis, creating the Podman network required for communications among containers.

Redhat users have different commands for root and rootless users.

【CASE Red Hat 1, CASE Red Hat 2】

// To create a network, use the following command
dfm network create

// Run the following command to see if “dfm-network” is visible.
dfm network ls

```
NETWORK ID          NAME          DRIVER          SCOPE
~~~~~~~~~~
e2697cd6621a        dfm-network   bridge          local
~~~~~~~~~~
```

【CASE Red Hat 3】

// To create a network, use the following command
dfm pod create

// Run the following command to see if “dfm-network” is visible.
dfm pod ls

```
POD ID          NAME       STATUS    CREATED      INFRA ID       # OF CONTAINERS
~~~~~~~~~~
df883e238ea9    dfm-pod    Running   9 days ago   97e26cc3bea3   1
~~~~~~~~~~
```

4.12. (STEP10) Start-up and Initializing the DFM Modules

We have created a service account and signed in using this account to create a service directory, install the DFM Module package, and load the Podman Image resources. The installer has now configured the resources and created the container to network. Now, start up the DFM Modules.

4.12.1. Start-up and Initialize MySQL Server (DFM DB)

In this stage, you will perform the following two steps:

1) Set the DB root password
2) Initialize the SQL query file copied in "4.3 Installing the DFM Module Package" above, on the mysql server

To do this, you must first start the mysql server container.

The command to run the mysql server container is as follows:

```
dfm start dfm-mysql
```

【Validation】

Make sure the MySQL container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-mysql
healthy
```

If the status is healthy, run each of the following commands.

1) Set DB root password: we assume that “password” is “**1q2w3e4r**”
We use this command: ALTER USER 'root'@'localhost' IDENTIFIED BY '{password}'

```
podman exec -it dfm-mysql mysql -uroot
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 5.7.25-log MySQL Community Server (GPL)

Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY '1q2w3e4r';
Query OK, 0 rows affected (0.00 sec)

mysql> exit
```

2) Run sql query file : we assume that password is “1q2w3e4r”

```
podman exec -i dfm-mysql mysql -uroot -p1q2w3e4r < /tmp/dfm/mysql-query/init_db.sql
mysql: [Warning] Using a password on the command line interface can be insecure.

podman exec -i dfm-mysql mysql -uroot -p1q2w3e4r <
/tmp/dfm/mysql- query/init_dfm_core.sql
mysql: [Warning] Using a password on the command line interface can be insecure.

podman exec -i dfm-mysql mysql -uroot -p1q2w3e4r <
/tmp/dfm/mysql- query/init_dfm_console.sql
mysql: [Warning] Using a password on the command line interface can be insecure.
```

```
$
```

4.12.2. Start-up Firmware Storage Server

In this stage, the installer starts the storage server that manages the firmware binary.
The command to run the Firmware Storage Server container is as follows:

```
dfm start dfm-minio
```

【Validation】

Make sure the Minio container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-minio
healthy
```

4.12.3. Start-up DFM Core Server

In this stage, the installer starts the DFM Core Server that provides the core API. The command to run the core server container is as follows:

```
dfm start dfm-core
```

【Validation】

Make sure the core container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-core
healthy
```

4.12.4. Start-up DFM Admin Console Server

In this stage, the installer starts the DFM Admin server that provides the admin console web page. The command to run admin server container is as follows:

```
dfm start dfm-console
```

【Validation】

Make sure the admin container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-console
healthy
```

4.12.5. Start-up HAProxy Server

In this step, the installer starts the HAProxy server that provides TLS/SSL termination and acts as the API gateway.

The command to run HAProxy server container is as follows:

```
dfm start dfm-proxy
```

【Validation】

Make sure HAProxy container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-proxy
healthy
```

4.13. How to check Server Operation Status

Finally, the installer has completed the installation of the on-premises service based Podman, and the service is now ready for use. However, we first need to validate whether the above five containers are running in a healthy state.

To check the status of the containers, use the command show below. If every status returns healthy, the service is ready for operation.

podman ps –a

Example)

podman ps –a

```
CONTAINER ID  IMAGE                                           COMMAND               CREATED     STATUS
97e26cc3bea3  registry.access.redhat.com/ubi8/pause:latest                           9 days ago  Up 9 days ago
87788c0f949a  localhost/mysql/enterprise-server:8.0            mysqld                9 days ago  Up 9 days ago
4f6bb6af2920  localhost/dfm-console:1.0.1.2-rootless                                 9 days ago  Up 9 days ago
636ab5081c12  localhost/dfm-core:1.0.1.2-rootless                                    9 days ago  Up 9 days ago
0f9bc568fcd5  localhost/minio/minio:RELEASE.2020-06-01T17-28-03Z  server /data          9 days ago  Up 9 days ago
af2c052532d5  localhost/haproxytech/haproxy-debian:2.1.4         haproxy -f /usr/l...  9 days ago  Up 9 days ago
```

In here, the health status means:

Healthy(0): Normal

Unhealthy(1): Abnormal

Starting (2): Starting

When the installer checks the health status after the installation is completed, if the status is not “Normal”, the installer must redo the installation. If the installation is unsuccessful after several tries, please contact the Samsung engineering team.

PART III: Initial Operation

PART III describes how to operate the Knox E-FOTA On-Premises service upon completion of the service installation on the customer's infrastructure.

5. Service Operation

This chapter explains how to check the operation status of each DFM Server, and how to use the service properly.

5.1. How to access to admin console page after installing

If you completed every installation step up to "***4.12.5. Start-up HAProxy Server***" in "***4.12. (STEP10) Start- up and Initializing the DFM Modules***", access the admin page to check whether the DFM Service is successfully installed and working.

【URL to the admin site】

{access_scheme}://{access_address}:{access_port}/admin/

⇒ Refer to "***4.9. (STEP07) Set-up Configuration***".

In this guide, we are using the URL and other information as follows:

- host_ip : 192.168.1.52
- listen_port : 80
- listen_scheme : http
- access_address : 181.107.61.233
- access_scheme : http
- access_port : 6380

【Account & Initial Password (PWD)】

⇒ Account will be: **admin**

⇒ Initial PWD will be: **admin12#**

*) *This PWD is created by Samsung, so **change the password** after you sign in.*

【Example】http://192.168.1.52:6380/admin/ (using a new **Chrome** browser)

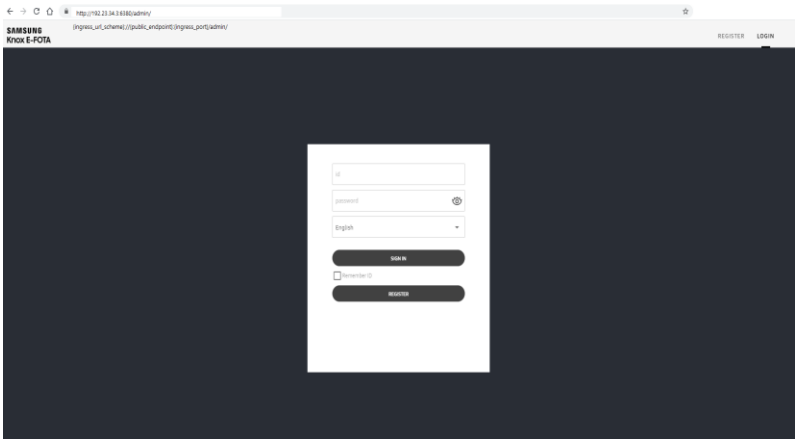

Fig 5-1 **The Admin Console for Knox E-FOTA On-Premises**

5.2. The Contents Upload

In order to use this service, IT admins must upload the contents (such as license and firmware) properly (please refer to the "**Knox E-FOTA On-Premises User Manual**" provided).

5.3. Troubleshooting and Logging during using the Service

While using this service, any issues should first be addressed on the site to avoid service disruptions from the issues. In order to support issue analysis, Samsung provides the "**TS & Logging Guide for Knox E-FOTA On-Premises**" guide for reference.

5.4. Updating the SSL Certificate when the old certificate is expired

SSL certificates have an expiration date. When the expiration date for the certificate approaches, the customer must reissue the certificate from the certificate signing authority before the current certificate expires.

There are two possibilities for TLS/SSL Termination.

1) On Customer's Load Balancer (proxy)
   We don't need to update the certificate file.
   Refer to (**[Use Case 2]:Type B**) in "*4.9. (STEP07) Set-up Configuration*"

2) On DFM Server
   We need to update the certificate file on the DFM Server.
   Refer to (**[Use Case 2]:Type C**) in "*4.9. (STEP07) Set-up Configuration*"

We assume that the new certificate file is "**new-example-fota.net.pem**", and we also assume that you are using the "**nightwatch**" service account.

【STEP01】Stop Proxy

The command to stop the proxy Server container is as follows:

```
dfm terminate dfm-proxy
```

【STEP02】Copy the new certificate

```
cp new-example-fota.net.pem /dfm/haproxy/config
sudo chown nightwatch:nightwatch /dfm/haproxy/config/new-example-fota.net.pem
sudo chmod 600 /dfm/haproxy/config/new-example-fota.net.pem
vi /dfm/haproxy/config/haproxy.cfg

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
frontend fe_web
```

```
    bind *:80
    bind *:443 ssl crt /usr/local/etc/haproxy/new-example-sec-fota.net.pem
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
backend dfmConsoleBackend
        mode http
        acl logout_path_set var(txn.path) path /admin/logout
        http-request set-header X-Forwarded-Port %[dst_port]
        http-request add-header X-Forwarded-Proto https if { ssl_fc }

        option httpchk GET /admin/health/live
        http-check expect status 200
        default-server inter 5s fall 3 rise 2

        # if DFM Server is behind customer's Load-Balancer and also customer's Load-Balancer provides ssl termination.
        #http-response replace-value Location (.*) https://%[var(txn.host)]/admin/ if logout_path_set
        # otherwise
        http-response replace-value Location (.*) %[var(txn.scheme)]://%[var(txn.host)]/admin/ if logout_path_set

       server dfm-console dfm-console:10050 check resolvers docker init-addr libc,none
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

【STEP03】Restart proxy

The command to restart the proxy Server container is as follows:

```
dfm start dfm-proxy
```

To make sure that the HAProxy container is in a healthy state, run the following command. It may take some time until the state shows as healthy.

```
podman healthcheck run dfm-proxy
healthy
```

5.5. DB Backup & Restore.

This section describes how to back up and restore the DB to ensure service continuity.

5.5.1. Back up a MySQL Server Instance

I. *To BACK UP a MySQL Server instance running in a Podman container using MySQL Enterprise Backup with Podman:*

1. On the same host where the MySQL Server container is running, start another container with an image of MySQL Enterprise Edition to perform a backup with the MySQL Enterprise Backup command "`backup-to-image`". Provide access to the server's data directory using the bind mount we created in the last step. Also, mount a host directory (*/path-on-host-machine/backups/* in this example) onto the storage folder for backups in the container (*/data/backups* in the example) to persist the backups we are creating.

Here is a sample command for this step. Here, we assume that 'root' account's password is **1q2w3e4r** and that the MySql podman image currently installed is *localhost/mysql/enterprise-server:8.0*.

We assume that you are using the "**nightwatch**" account.

【Basic Command】

【CASE Red Hat 1】

```
podman run \
   --mount  type=bind,src=/path-on-host-machine/datadir/,dst=/var/lib/mysql  \
   --mount type=bind,src=/path-on-host-machine/backups/,dst=/data/backups \
   --rm localhost/mysql/enterprise-server:8.0 \
   mysqlbackup -u{user} -p{password} --backup-dir=/tmp/backup-tmp --with-timestamp \
   --backup-image=/data/backups/{db-file-name}.mbi   backup-to-image
```

【 CASE Red Hat 2, CASE Red Hat 3】

```
podman unshare chown -R nightwatch:nightwatch {/path-on-host-machine/backups}


podman run \
    --security-opt label=type:dfm_mysql_t.process\
   --mount  type=bind,src=/path-on-host-machine/datadir/,dst=/var/lib/mysql  \
   --mount type=bind,src=/path-on-host-machine/backups/,dst=/data/backups \
   --rm localhost/mysql/enterprise-server:8.0 \
   mysqlbackup -u{user} -p{password} --backup-dir=/tmp/backup-tmp --with-timestamp \
   --backup-image=/data/backups/{db-file-name}.mbi   backup-to-image
```

【Example】

```
podman unshare chown -R nightwatch:nightwatch /dfm/mysql/backups

podman run \
  --security-opt label=type:dfm_mysql_t.process\
  --mount  type=bind,src=/dfm/mysql/data,dst=/var/lib/mysql  \
  --mount  type=bind,src=/dfm/mysql/backups,dst=/data/backups  \
  --rm  localhost/mysql/enterprise-server:8.0  \
  mysqlbackup  -uroot  -p1q2w3e4r  --backup-dir=/tmp/backup-tmp  --with-timestamp  \
  --backup-image=/data/backups/db-2020-05-20.mbi     backup-to-image

  [Entrypoint] MySQL Docker Image 8.0.20-1.1.16
  MySQL Enterprise Backup Ver 8.0.20-commercial for Linux on x86_64 (MySQL Enterprise - Commercial)
  Copyright (c) 2003, 2020, Oracle and/or its affiliates. All rights reserved.

  Oracle is a registered trademark of Oracle Corporation and/or its
  affiliates. Other names may be trademarks of their respective
  owners.

  Starting with following command line ...
    mysqlbackup -uroot -pxxxxxxxxx --backup-dir=/tmp/backups --with-timestamp
            --backup-image=/data/backups/db-2020-05-20.mbi  backup-to-image
```

```
                At the end of a successful 'backup-to-image' run mysqlbackup
                prints "mysqlbackup completed OK!".

  200520 01:40:40 MAIN     INFO: Starting to log actions.
  200520 01:40:40 MAIN     INFO: No SSL options specified.
  .....

  200520 01:40:42 MAIN     INFO: Backup image created successfully.
  200520 01:40:42 MAIN     INFO: Image Path = /data/backups/db-2020-05-20.mbi
  200520 01:40:42 MAIN     INFO: MySQL binlog position: filename binlog.000005, position 530056

  ------------------------------------------------------------------------
      Parameters Summary
  ------------------------------------------------------------------------
      Start LSN                    : 32781824
      End LSN                      : 32848770
  ------------------------------------------------------------------------

  mysqlbackup completed OK!
```

It is important to check the end of the output next to **mysqlbackup** to make sure the backup has been completed successfully.

2. The container exits once the backup job is finished and, with the *--rm* option used to start it, it is removed after it exits. An image backup is created and can be found in the host directory mounted in the last step for storing backups:

```
ls /dfm/mysql/backups/
db-2020-05-20.mbi
```

5.5.2. Restore a MySQL Server Instance

II. *To RESTORE a MySQL Server instance in a Podman container using MySQL Enterprise Backup with Podman:*

1. Stop the MySQL Server container, which also stops the MySQL Server runninginside:
mounted in the last step for storing backups:

```
docker stop dfm-mysql
```

2. On the host, delete all contents in the bind mount for the MySQL Server data directory:

```
sudo rm -rf /dfm/mysql/data/*
```

3. Start a container with an image of MySQL Enterprise Edition to perform the restore with the MySQL Enterprise Backup command copy-back-and-apply-log. Bind-mount the server's data directory and the storage folder for the backups, like what we did when we backed up the server:

【Basic Command】

【CASE Red Hat 1】

```
podman run \
  --mount  type=bind,src=/path-on-host-machine/datadir/,dst=/var/lib/mysql  \
  --mount type=bind,src=/path-on-host-machine/backups/,dst=/data/backups \
  --rm localhost/mysql/enterprise-server:8.0 \
  mysqlbackup --backup-dir=/tmp/backup-tmp --with-timestamp \
  --datadir=/var/lib/mysql/  --backup-imag=/data/backups/{db-file-name-to-restore}.mbi  copy-back-and-apply-log
```

【 CASE Red Hat 2, CASE Red Hat 3】

```
podman run \
  --security-opt label=type:dfm_mysql_t.process\
  --mount  type=bind,src=/path-on-host-machine/datadir/,dst=/var/lib/mysql  \
  --mount type=bind,src=/path-on-host-machine/backups/,dst=/data/backups \
  --rm localhost/mysql/enterprise-server:8.0 \
  mysqlbackup --backup-dir=/tmp/backup-tmp --with-timestamp \
  --datadir=/var/lib/mysql/  --backup-imag=/data/backups/{db-file-name-to-restore}.mbi  copy-back-and-apply-log
```

【Example】

```
podman run \
      --security-opt label=type:dfm_mysql_t.process\
      --mount  type=bind,src=/dfm/mysql/data,dst=/var/lib/mysql  \
      --mount type=bind,src=/dfm/mysql/backups,dst=/data/backups \
      --rm localhost/mysql/enterprise-server:8.0 \
      mysqlbackup --backup-dir=/tmp/backup-tmp --with-timestamp \
      --datadir=/var/lib/mysql   --backup-image=/data/backups/db-2020-05-20.mbi   copy-back-and-apply-log


   MySQL Enterprise Backup Ver 8.0.20-commercial for Linux on x86_64 (MySQL Enterprise - Commercial)
   Copyright (c) 2003, 2020, Oracle and/or its affiliates. All rights reserved.

   Oracle is a registered trademark of Oracle Corporation and/or its
   affiliates. Other names may be trademarks of their respective
   owners.

   Starting with following command line ...
     mysqlbackup -uroot -pxxxxxxxx --backup-dir=/tmp/backup-tmp
                --with-timestamp --datadir=/var/lib/mysql
                --backup-image=/data/backups/db-2020-05-20.mbi    copy-back-and-apply-log

   IMPORTANT: Please check that mysqlbackup run completes successfully.
                    At the end of a successful 'copy-back-and-apply-log' run mysqlbackup
                    prints "mysqlbackup completed OK!".

   200520 02:01:08 MAIN        INFO: Starting to log actions.
   [Entrypoint] MySQL Docker Image 8.0.20-1.1.16
   .......
   200520 02:01:10 PCR1        INFO: We were able to parse ibbackup_logfile up to
                  lsn 32848770.
   200520 02:01:10 PCR1        INFO: Last MySQL binlog file position 0 530056, file name binlog.000005
   200520 02:01:10 PCR1        INFO: The first data file is '/var/lib/mysql/ibdata1'
                                        and the new created log files are at '/var/lib/mysql'
   200520 02:01:10 MAIN        INFO: Apply-log operation completed successfully.
   200520 02:01:10 MAIN        INFO: Full Backup has been restored successfully.

   mysqlbackup completed OK! with 3 warnings
```

The container exits once the backup job is finished and, with the *--rm* option used when starting it, it is removed after it exits.

4. Restart the server container, which also restarts the restored server:

```
podman restart dfm-mysql
```

6. When a Server is Re-booted

This chapter explains the steps to restart the DFM Modules if the server is rebooted, to ensure the service can run properly.

The steps to start the DFM Module server are as follows:

6.1. (STEP01) Log in as the dedicated service account

The DFM Module is logged in with a dedicated service account and operates with the privileges of the account (see, "*4.2. (STEP01) Create Service Account and Login*").

6.2. (STEP02) Prepare "mount" for DFM modules

The DFM module is installed and operates in the below directory on the **dedicated disk**.

The customer **may NOT configure** the auto-mount on the dedicated disk. For such cases, it is necessary to manually mount the dedicated disk on **/dfm**.

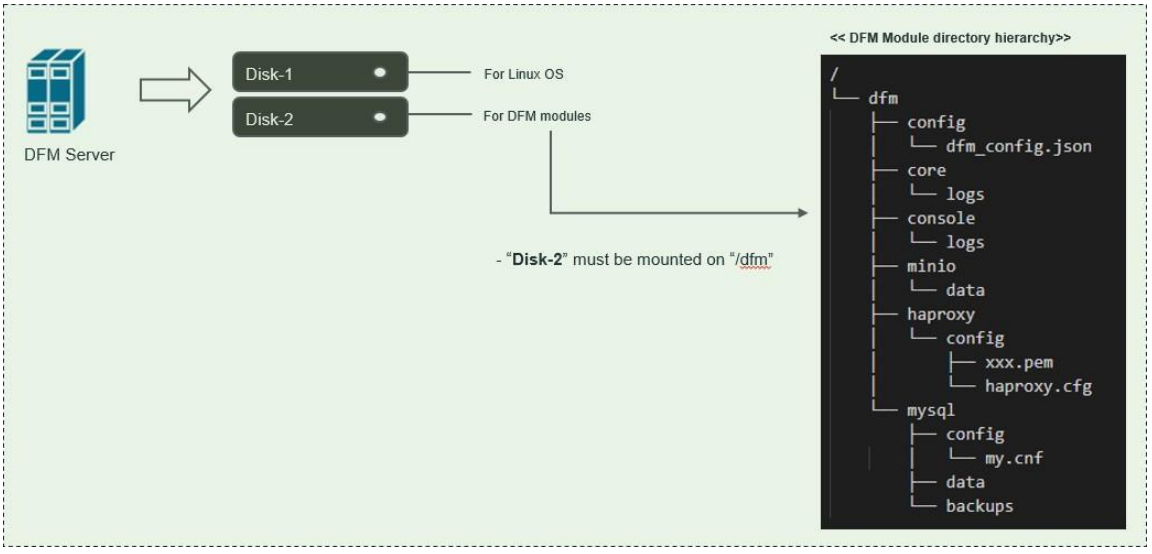

Fig 6-1 **A dedicated disk for DFM module**

For example, we assume that two disks ("sda" and "sdb") exist.

【CASE01】Disk is Ready

If the disk is ready, we don't need to mount it. Now, let's check the disk information:

```
sudo lsblk –p
```

```
NAME         MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
/dev/sda     202:0    0    1T  0 disk
└─/dev/sda1 202:1    0    1T  0 part /
/dev/sdb     202:80   0    1T  0 disk
```

sudo lsblk –f

```
NAME    FSTYPE   LABEL          UUID                                   MOUNTPOINT
sda
└─sda1  ext4     xxxxxxxx-rootfs 6156ec80-9446-4eb1-95e0-9ae6b7a46187 /
sdb     ext4                     d3269ceb-4418-45d0-ba68-d6b906e0595d /dfm
```

⇨ “sdb” is already formatted and mounted on **/dfm**

sudo file –s /dev/sdb

```
/dev/sdb: Linux rev 1.0 ext4 filesystem data, UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d (extents) (64bit) (large files) (huge files)
```

【CASE02】Disk is NOT Ready : it is already formatted but Not yet mounted on /dfm

If the disk is formatted but not yet mounted, it needs to be mounted on **/dfm**. Now, let’s check the disk information.

sudo lsblk –p

```
NAME         MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
/dev/sda     202:0    0    1T  0 disk
└─/dev/sda1 202:1    0    1T  0 part /
/dev/sdb     202:80   0    1T  0 disk
```

sudo lsblk –f

```
NAME    FSTYPE   LABEL          UUID                                   MOUNTPOINT
sda
└─sda1  ext4     xxxxxxxx-rootfs 6156ec80-9446-4eb1-95e0-9ae6b7a46187 /
sdb     ext4                     d3269ceb-4418-45d0-ba68-d6b906e0595d
```

⇨ “sdb” ” is formatted but not yet mounted

1) Mount /dev/sdb on /dfm

```
// create directory to mount
sudo mkdir /dfm

// mount
sudo mount /dev/sdb /dfm
```

2) Verify

```
df -h
```

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb        9.8G   37M  9.3G   1% /dfm
```

6.3. (STEP03) Start up Database Server (MySQL)

After the system is rebooted, restart MySQL using the following command:

```
dfm restart dfm-mysql
```

【Validation】

Run the following command to ensure the MySOL container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-mysql
healthy
```

6.4. (STEP05) Start up Firmware Storage Server

After the system is rebooted, restart Minio. The command to run the Minio server container is as follows:

```
dfm restart dfm-minio
```

【Validation】

Run the following command to make sure the Minio container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-minio
healthy
```

6.5. (STEP06) Start up DFM Core Server

After the system is rebooted, restart DFM Core. The command to run the core server container is as follows:

```
dfm restart dfm-core
```

【Validation】

Run the following command to make sure the core container is in a healthy state. It takes some time until its state is healthy.

```
podman healthcheck run dfm-core
healthy
```

6.6. (STEP07) Start up DFM Admin Console Server

After the system is rebooted, restart DFM Admin. The command to run the admin server container is as follows:

```
dfm restart dfm-console
```

【Validation】

Run the following command to ensure the admin container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-console
healthy
```

6.7. (STEP08) Start up HAProxy Server

After the system is rebooted, restart HAProxy. The command to run the HAProxy server container is as follows:

```
dfm restart dfm-proxy
```

【Validation】

Run the following command to ensure HAProxy container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-proxy
healthy
```

6.8. (STEP09) Check Server Operation Status

Finally, once every resource is restarted, their states must be verified as healthy. The administrator can use the following command to do so.

If the status of all containers show as healthy, the platform is running normally.

podman ps –a

Example)

podman ps –a

```
CONTAINER ID  IMAGE                                                COMMAND               CREATED     STATUS
97e26cc3bea3  registry.access.redhat.com/ubi8/pause:latest                               9 days ago  Up 9 days ago
87788c0f949a  localhost/mysql/enterprise-server:8.0                 mysqld                9 days ago  Up 9 days ago
4f6bb6af2920  localhost/dfm-console:1.0.1.2-rootless                                     9 days ago  Up 9 days ago
636ab5081c12  localhost/dfm-core:1.0.1.2-rootless                                        9 days ago  Up 9 days ago
0f9bc568fcd5  localhost/minio/minio:RELEASE.2020-06-01T17-28-03Z   server /data          9 days ago  Up 9 days ago
af2c052532d5  localhost/haproxytech/haproxy-debian:2.1.4            haproxy -f /usr/l...  9 days ago  Up 9 days ago
```

PART IV: Update the DFM Modules

PART IV: Update the DFM Modules describes how to update the Knox E-FOTA On-Premises service if there are any updates within the service resources.

7. Update the DFM Module

This chapter explains how to update the DFM Modules in operation, such as a fetch version. In order to properly update each module, the updater must first stop the module based on the related command (see, ***Appendix B***).

During the update, the Knox E-FOTA On-Premises service may not be available.

The DFM Module is logged in with a dedicated service account and operates with the privileges of the account. Ensure you log in with the account you previously used for installation.

7.1. Podman Image Update

If there is an updated DFM Module, it is also released as a Podman Image Package and packed as a tar file. In the release, the Podman Image contains repository and tag information as well.

7.1.1. DFM Database Update (MySQL)

For example, assume that the released **MySQL** image information is as follows:

- Podman image: dfm-mysql-xx.xx.xx.tar
- repository: dfm-mysql
- tag: xx.xx.xx

It should be updated as per the following steps:

【**STEP01**】Stop the running DFM Core Server, Admin Console Server, and Mysql Server.

```
dfm terminate dfm-core
dfm terminate dfm-console
dfm terminate dfm-mysql
```

【**STEP02**】Load the released Podman Image.

```
podman load -i dfm-mysql-xx.xx.xx.tar
```

【**STEP03**】Change the repository and tag's configuration

```
dfm config set mysql_img_rep=dfm-mysq"
dfm config set mysql_img_tag=xx.xx.xx
```

【**STEP04**】Confirm the changed repository and tag's configuration

```
dfm config get mysql_img_rep
dfm config get mysql_img_tag
```

【**STEP05**】Start up Server

- MySQL Server

```
dfm start dfm-mysql
```

【Validation】

Run the following command to ensure the mysql container is in a healthy state. It takes some time until its state is healthy.

```
podman healthcheck run dfm-mysql

podman ps -a
```

7.1.2. DFM Firmware Storage Update (MinIO)

For example, assume that the released **MinIO** image information is as follows:

- Podman image : dfm-minio-xx.xx.xx.tar
- repository : dfm-minio
- tag : xx.xx.xx

【**STEP01**】Stop the MinIO server.

```
dfm terminate dfm-minio
```

【**STEP02**】Load the released Podman Image.

```
podman load -i dfm-minio-xx.xx.xx.tar
```

【**STEP03**】Change the repository and tag's configuration

```
dfm config set minio_img_rep=dfm-minio
dfm config set minio_img_tag=xx.xx.xx
```

【**STEP04**】Confirm the changed repository and tag's configuration

```
dfm config get minio_img_rep
dfm config get minio_img_tag
```

【**STEP05**】Start-up Server

- MinIO Server

```
dfm start dfm-minio
```

【Validation】
Run the following command to ensure the mysql container is in a healthy state. It takes some time until its state is healthy.

```
podman healthcheck run dfm-minio

podman ps -a
```

7.1.3. DFM Core Update

For example, assume that the released **Core** image information is as follows:

- Podman image : dfm-core-xx.xx.xx.tar
- repository : dfm-core
- tag : xx.xx.xx

【**STEP01**】Stop the running core server.

```
dfm terminate dfm-core
```

【**STEP02**】Load the released Podman image.

```
podman load -i dfm-core-xx.xx.xx.tar
```

【**STEP03**】Change the repository and tag's configuration

```
dfm config set core_img_rep=dfm-core
dfm config set core_img_tag=xx.xx.xx
```

【STEP04】Confirm the changed repository and tag's configuration

```
dfm config get core_img_rep
dfm config get core_img_tag
```

【STEP05】Start-up Server

- DFM Core Server

```
dfm start dfm-core
```

【Validation】
Run the following command to ensure the mysql container is in a healthy state. It takes some time until its state is healthy.

```
podman healthcheck run dfm-core

podman ps -a
```

7.1.4. DFM Admin Console Update

For example, assume that the released **Admin** image information is as follows:

- Podman image : dfm-console-xx.xx.xx.tar
- repository : dfm-console
- tag : xx.xx.xx

【STEP01】Stop the running core, admin and mysql servers.

```
dfm terminate dfm-console
```

【STEP02】Load the released Podman image.

```
podman load -i dfm-console-xx.xx.xx.tar
```

【STEP03】Change the repository and tag's configuration

```
dfm config set console_img_rep=dfm-console
dfm config set console_img_tag=xx.xx.xx
```

【STEP04】Confirm the changed repository and tag's configuration

```
dfm config get console_img_rep
dfm config get console_img_tag
```

【STEP05】Start-up Server

- Admin Console Server

```
dfm start dfm-console
```

【Validation】
Run the following command to make sure the mysql container is in a healthy state. It takes some time until its state is healthy.

```
podman healthcheck run dfm-console

podman ps -a
```

7.1.5. HAProxy update

For example, assume that the released **HAProxy** image information is as follows:

- Podman image : dfm-haproxy-xx.xx.xx.tar
- repository : dfm-haproxy
- tag : xx.xx.xx

【**STEP01**】Stop the running haproxy server.

```
dfm terminate dfm-proxy
```

【**STEP02**】Load the released Podman image.

```
podman load -i dfm-haproxy-xx.xx.xx.tar
```

【**STEP03**】Change the repository and tag's configuration

```
dfm config set haproxy_img_rep=dfm-haproxy
dfm config set haproxy_img_tag=xx.xx.xx
```

【**STEP04**】Confirm the changed repository and tag's configuration

```
dfm config get haproxy_img_rep
dfm config get haproxy_img_tag
```

【**STEP05**】Start-up Server

- HAProxy Server

```
dfm start dfm-proxy
```

【Validation】
Run the following command to ensure the HAProxy container is in a healthy state. It may take some time until its state is healthy.

```
podman healthcheck run dfm-proxy

podman ps -a
```

7.2. The Contents Update

In order to use this service, IT admins must upload the contents (such as license and firmware) properly (please refer to the "**Knox E-FOTA On-Premises User Manual**" provided).

PART V: Purge DFM Modules

This section, which covers purging the DFM Modules, describes how to erase all installed services when you want to delete the existing installed modules.

Please note that doing so **erases all existing data**.

After completing these actions, you can reinstall the DFM modules without any interference from the old installation (see 4.3. (STEP03) Create Service Directories).

8. Purge the DFM Modules

This chapter explains how to purge the installed DFM Modules.

The DFM Module is logged in with a dedicated service account and operates with the privileges of the account. Log in with the account you used during the installation.

8.1. Terminate Services

If there are active services, terminate them.

【**STEP01**】Check if there are any running or exited services. If they exist, we need to terminate them.

podman ps –a

Example)

podman ps –a

```
CONTAINER ID  IMAGE                                           COMMAND               CREATED     STATUS
97e26cc3bea3  registry.access.redhat.com/ubi8/pause:latest                           9 days ago  Up 9 days ago
87788c0f949a  localhost/mysql/enterprise-server:8.0            mysqld                9 days ago  Up 9 days ago
4f6bb6af2920  localhost/dfm-console:1.0.1.2-rootless                                 9 days ago  Up 9 days ago
636ab5081c12  localhost/dfm-core:1.0.1.2-rootless                                    9 days ago  Up 9 days ago
0f9bc568fcd5  localhost/minio/minio:RELEASE.2020-06-01T17-28-03Z  server /data          9 days ago  Up 9 days ago
af2c052532d5  localhost/haproxytech/haproxy-debian:2.1.4         haproxy -f /usr/l...  9 days ago  Up 9 days ago
```

1. DFM Database (MySQL)

   Stop the server with the following command:

   ```
   dfm terminate dfm-mysql
   ```

2. DFM Firmware Storage (MinIO)

   Stop the server with the following command:

   ```
   dfm terminate dfm-minio
   ```

3. DFM Core Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-core
   ```

4. DFM Admin Console Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-console
   ```

5. DFM HAProxy Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-proxy
   ```

6. Check if all services are removed.

   Check with the following command:

   ```
   ps -a
   ```

8.2. Remove Service directory

Remove old data using the following:

Remove all directory in /dfm

```
cd /dfm
sudo rm -rf *
```

PART VI: APPENDICES

PART IV: APPENDICES presents more in-depth explanations for each item.

APPENDICES

Appendix A. Terms and Abbreviations

This chapter outlines the terms and abbreviations used in this guide.

App: Application
CAT: Category Codes
CSO/TEO: Customer Service Operation/Technical Engineer for On-Premise
CM: Commercial Type Product
DE: Docker Enterprise
DFM: Device Firmware Management
DNS: Domain Name Server
E2E: End to End
E-FOTA: Enterprise – Firmware over the Air
FYI: For Your Information
HA: High Availability
H/W: Hardware
ID: Identification
KE: Knox E-FOTA (Brand)
LB: Load Balancer
NAT: Network Address Translation
OS: Operating System
PoC: Proof of Concept
PWD: Password
SSL: Secure Sockets Layer
TLS: Transport Layer Security, successor to SSL
UI: User Interface

Appendix B. How to terminate each DFM Module

These commands should not be used in normal operation, as stopping a module can seriously impact how the service runs. Use this command for updates, such as when there is a fetch version delivery.

1. DFM Database (MySQL)

   Stop the server with the following command:

   ```
   dfm terminate dfm-mysql
   ```

2. DFM Firmware Storage (MinIO)

   Stop the server with the following command:

   ```
   dfm terminate dfm-minio
   ```

3. DFM Core Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-core
   ```

4. DFM Admin Console Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-console
   ```

5. DFM HAProxy Server

   Stop the server with the following command:

   ```
   dfm terminate dfm-proxy
   ```

Appendix C. Summary for Software (S/W) Recommendation

Read more about detailed recommendations in "*2.3. Recommendation Per each Product usage*".

Product	Category	S/W	Version	Supported Options	Additional Info
CM	**Server OS**	Ubuntu	18.04.3 LTS	Enterprise (**Paid**)	https://assets.ubuntu.com/v1/1a8fb1b3-UA-I_datasheet_2019-Oct.pdf?_ga=2.267414477.2124202676.1591159591-176408230.1591159591
	Container	Docker Engine	Community Edition	Community (Free)	https://info.mirantis.com/l/530892/2018-04-12/37s6c/530892/93926/Mirantis_Support_Subscription_Brochure.pdf
	Database	MySQL	Enterprise Edition	Enterprise (**Paid**)	https://www.mysql.com/products/
PoC	**Server OS**	Ubuntu	18.04.3 LTS	Community (free)	
	Container	Docker Engine	Community Edition	Community (Free)	
	Database	MySQL	Community Edition	Community (Free)	If a customer wants to continue using the Commercial (**CM**) product after PoC ends, recommend **Enterprise Edition** for both **Server OS** and **Database** at the start of the PoC

Appendix D. A Recommended Schedule for On-Site Installation by CSO/TEO

This recommended schedule can be used by the CSO/TEO while they are doing the on-site installation. The detailed schedule can be freely modified.

We recommend "The 4-Day Installation", as the customer should understand how they are using the Knox E-FOTA On-Premises service during this program. A training session should be included to support this purpose as well.

Day	Actions	Program
Day1	Check the customer's infrastructures (such as H/W and S/W) to install the service on, based on the prerequisites (see "*2.3 Recommendation Per each Product usage*")	1. Introduce each other 2. Introduce "The 4-Days Installation" program 3. Introduce the Knox E-FOTA On-Premises service (using "**KE On-Premise Service Intro 2020.pdf**" 4. Check the customer's infrastructures 1) H/W recommendation, such as Server CPU cores, RAM, Disk, Network Card 2) S/W recommendation, such as Operating System, Docker Engine, MySQL Edition, and whether those have been installed by the customer 3) Get public certificate files for https 4) Get port number (6443) for https 5. Wrap-up
Day2	Perform the installation based on this guide (see "*4. Installation & Configuration*")	1. Introduce the program to install 2. Start Installation 3. Configure the DFM service infrastructure 4. Check the service operation via the Web Console 5. Wrap-up
Day3	Perform an acceptance test through E2E with devices	1. Introduce how to do an E2E test with devices 2. Introduce how to use the service Web Console (using "**Knox E-FOTA On-Premises User Guide.pdf**, and **Knox E-FOTA On-Premises User Guide for Device.pdf**") 3. Upload the License onto the Server 4. Upload the Firmware deltas (Contents for FOTA) 5. Upload the device information used during the test 6. Create the Campaign 7. Do E2E test with devices 8. Wrap-up
Day4	Introduce Operation and Maintenance procedures (Get document for "**The Confirmation of Installation Process End**" from the Customer)	1. Introduce the steps and how to perform them if there is an issue ☜ Using "**TS & Logging Guide for Knox E-FOTA On-Premise.pdf**" 2. Introduce how to raise issues ☜ Using "**Issue raising process**" 3. Introduce service operation steps 4. ☜ Using "**Service Operation Guide**" 5. Sign the "Notice for Completion Installation" ☜ Refer to "**Appendix E**" (*Installation and Initial Operation Guide for Knox E-FOTA On-Premises.pdf*) 6. Wrap-up

Appendix E. An Example of "Notice for Completion Installation"

Notice for Completion Installation

Dear < Customer Name >,

This form is to sign-off completion of your project with us. Kindly complete as best as possible and send back to us.

PRODUCT: Knox E-FOTA One On-premise	MANAGER NAME:
START DATE:	COMPLETION DATE:
June 1 2020 ~ June 4 2020	

DELIVERABLES:

1. **Device Client**
   It means Client application running on Samsung mobile devices. It is responsible for interacting with the E-FOTA (Enterprise-Firmware Over The Air) Server, including binary package download, and installer activation for the binary package.

2. **Device Firmware Management (DFM)**
   It is a main module for E-FOTA, including managed devices to FOTA, creation and management of FOTA Campaigns, and Firmware binaries for devices.
   It is consist of followings:
   1) DFM Core – It consists of Core Backend and Front End for Administrators
   2) DB (MySQL) – It is a data base for system operation
   3) Storage – It is a storage for Firmware binaries

3. **Installed in Customer's Environment**
   It depends on the contraction.
   1) Pre-Prod Environment (1 Set)
   2) Prod Environment (1 Set)

CUSTOMER'S COMMENTS:

REMARK:

By signing this document, I acknowledge that I have delivered all the stated deliverables.	By signing this document, I acknowledge that I have received all the stated deliverables.
Samsung (subsidiary office name)	**< Customer Name >**
Name:_____	Name:_____
Signature:_____	Signature:_____
Date: _____	Date: _____

We recommend that you complete and send this form within 5 working days. However, if after this period we do not receive the completed form, we shall assume that the project has been signed off by you and no further action will be required of you.

< EOF (End Of File) >