SAMSUNG ELECTRONICS

Knox E-FOTA On-Premises

Installation and Initial Operation Guide

Version : 1.6

Last Update : Dec 2025

【Document History】

What	*Ver.*	*When*
I. Updated to 1.0.1.11	1.6	Dec 2025
II. Updated 2.2. Software Recommended Appendix C. Summary for Software (S/W) Recommendation	Ver1.5	May 2025
III. **Added:** Appendix F. Set E-FOTA agent config by managed Configuration IV. **Updated:** 4.2.1) Arguments HTTPS Handling <- added how to make pem file for device	Ver1.4	Dec 2024
I. Added: 6.4. (STEP04) Start-up Background app 8.2.7 Stop background app and check	Ver1.3	Jun 2024
I. Added: 5.8 Configurable device polling interval and postpone waiting time	Ver1.2	Oct 2023
I.Added: 5.7 Configurable device group polling	Ver1.1	Apr 2023
Initial Release	Ver1.0	Nov 2022

Table of Contents

Tables of Figures & Tables

【Figures】

【Tables】

PART I: Getting Started

PART 1: Getting Started presents the purpose of this document, what customer infrastructure is recommended prior to the installation of the Knox E-FOTA On-Premises service, and provides an overview of deliverables that will 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 the 3rd party software, such as Docker, 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**.

In order 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" 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-Premises service by Samsung personnel.

2.1. Hardware Recommended

The recommended user environment, including the network card, for the On-Premises Hardware (H/W) requirements are as follows (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 Hardware Recommended for the Knox E-FOTA On-Premises user work environment

The recommendations in the above table are the minimum specifications to run this On-Premises Service. User performance expectations may require additional infrastructure resources that exceed the minimum specifications.

2.2. Software Recommended

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

Items	Recommended Value	Description
Operating System	Ubuntu Server 18.04.3 LTS, 22.04.4 LTS, or 24.04 LTS	
Docker Engine	Community Edition (Ubuntu)	
MySQL Edition	Community Edition	For Commercial Product

Table 2-2 The Software Recommended for the Knox E-FOTA On-Premises user work environment

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

2.2.1. Operating System

By default, the DFM Server requires Ubuntu Server 18.04.3 LTS, 22.04.4 LTS, or 24.04 LTS for the OS. It should be installed on64-bit Intel x86, ARM, or MIPS architectures in order to support Docker.

2.2.2. Docker

Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all of the parts it needs, such as libraries and other dependencies, and deploy it as one package. By doing so, thanks to the container, the application will run on any other Linux machine regardless of any customized settings that machine might have that could differ from the machine used for writing and testing the code.

In a way, Docker is like a virtual machine. Unlike a virtual machine, however, rather than creating a whole virtual operating system, Docker allows applications to use the same Linux kernel as the system it’s running on and only requires applications be shipped with things not already running on the host computer. This provides a significant performance boost and reduces the size of the application. In this On-Premises service, **the Community version** for Ubuntu will be using Docker. This version can be downloaded from “**download.docker.com**”.

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-premises 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 H/W 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

【S/W Recommendation】

Items	Recommended Value	Description
Operating System	Ubuntu Server 18.04.3 LTS, 22.04.4 LTS, or 24.04 LTS	
Docker Engine	Community Edition (Ubuntu)	
MySQL Edition	Community Edition	

Table 2-4 The Software Recommendation for “PoC”

The customer can purchase Ubuntu 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.

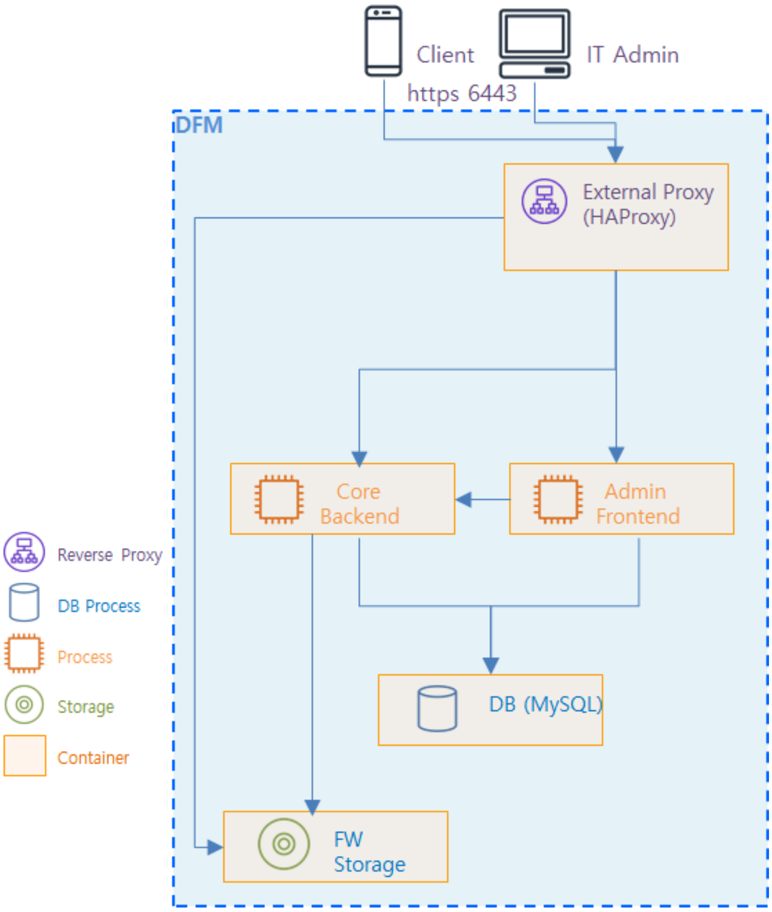


Fig 2-1 Knox E-FOTA On-Premises Product Arch for PoC

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 H/W 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”

【S/W Recommendation】

The customer can purchase Ubuntu OS based 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	Ubuntu Server 18.04.3 LTS, 22.04.4 LTS, or 24.04 LTS	
Docker Engine	Community Edition (Ubuntu)	
MySQL Edition	Community Edition	

Table 2-6 Software Recommendation for “Commercial”

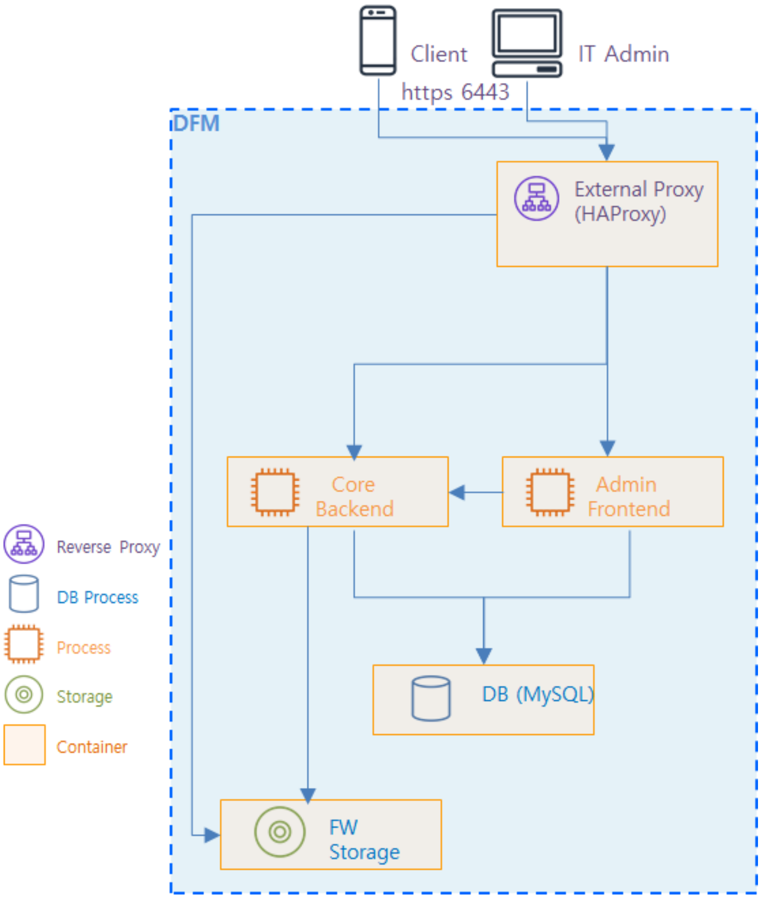


Fig 2-2 Knox E-FOTA On-Premises Product Arch for “Commercial”

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:** This is used for redirection between outer and DFM modules, and for AP Gateway and TLS/SSL termination

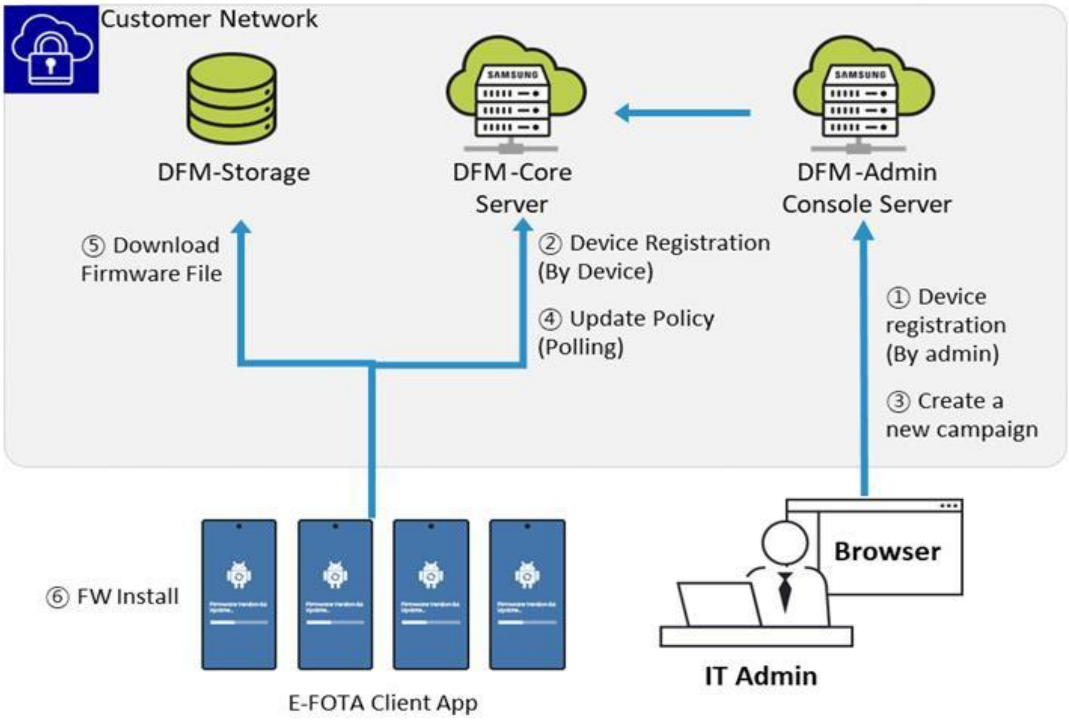


Fig 3-1 Knox E-FOTA On-Premises Conceptual Architecture

3.2. Security Considerations

In order 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.

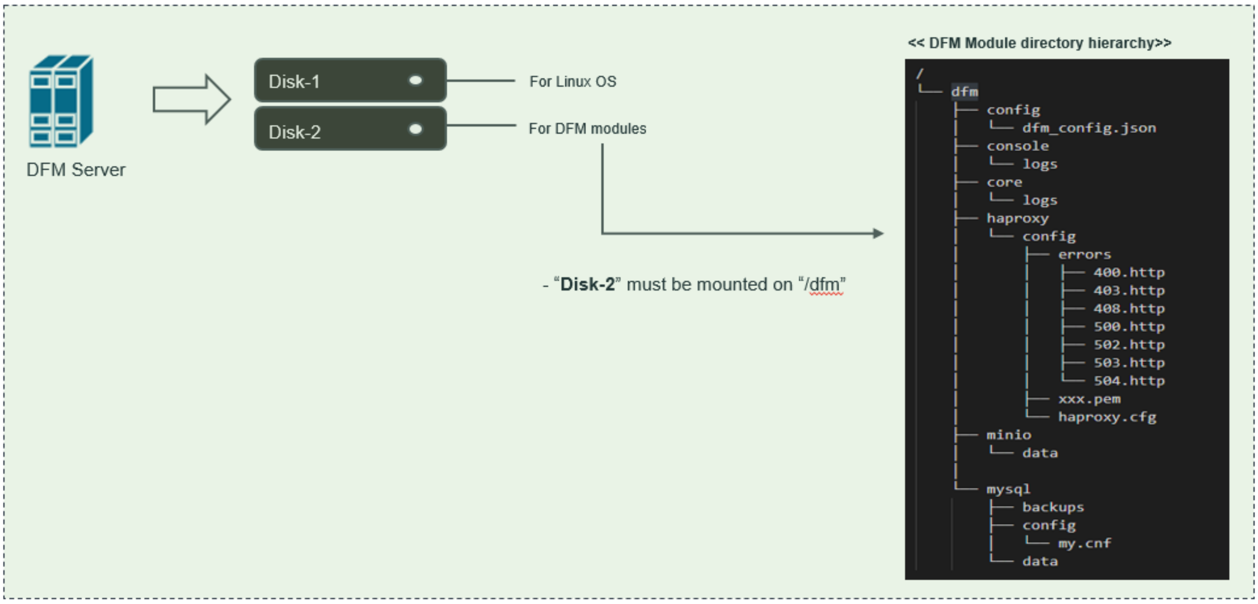
4. Installation & Configuration

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.

4.1. Prepare “Disk partition & mount” for DFM modules

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

Therefore, we should 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 DFM module before.


DFM Server
Disk-1
Disk-2
For Linux OS
For DFM modules
- “Disk-2” must be mounted on “/dfm”
<< DFM Module directory hierarchy>>

```
/
└── dfm
    ├── config
    │   └── dfm_config.json
    ├── console
    │   └── logs
    ├── core
    │   └── logs
    ├── haproxy
    │   └── config
    │       ├── errors
    │       │   ├── 400.http
    │       │   ├── 403.http
    │       │   ├── 408.http
    │       │   ├── 500.http
    │       │   ├── 502.http
    │       │   ├── 503.http
    │       │   └── 504.http
    │       ├── xxx.pem
    │       └── haproxy.cfg
    ├── minio
    │   └── data
    │
    └── mysql
        ├── backups
        ├── config
        │   └── my.cnf
        └── data
```

Fig 4-1 An Disk Partitions for DMF Module

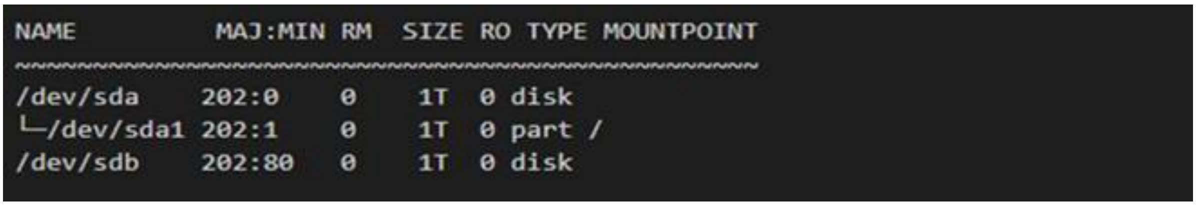
For example, we assume that two disks (“sda” and “sdb”) exist.

【CASE01】 Disk is Ready

If the disks exist, we don’t need to format and mount them.

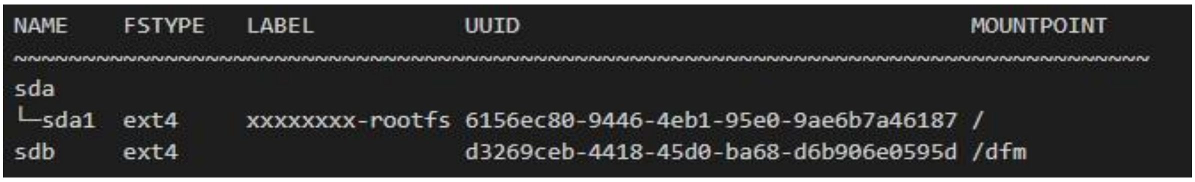
Now, let’s check the disk information:

```
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
```

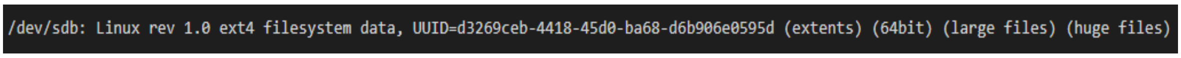
```
sudo lsblk -f
```



```
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
```



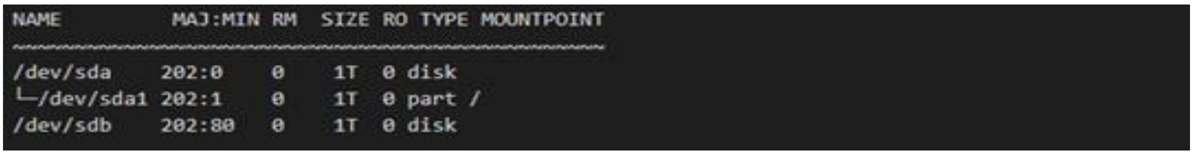
```
/dev/sdb: Linux rev 1.0 ext4 filesystem data, UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d (extents) (64bit) (large files) (huge files)
```

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

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

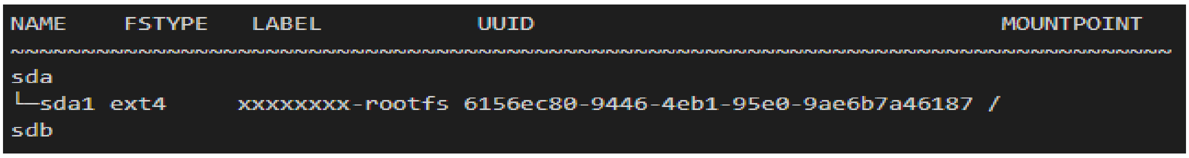
Now, let’s check the disk information:

```
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
```

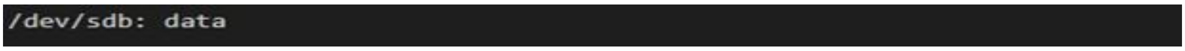
```
sudo lsblk -f
```



```
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
```

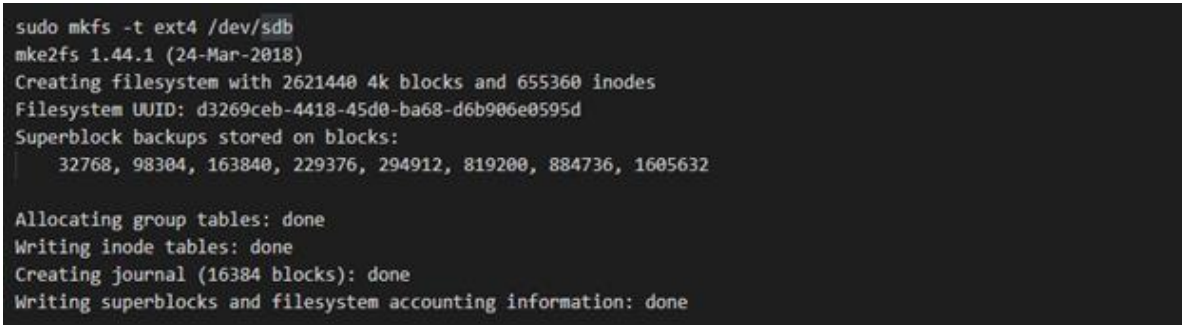


```
/dev/sdb: data
```

⇨ This means that the disk needs to be formatted

1) Format with ext4 file-system

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

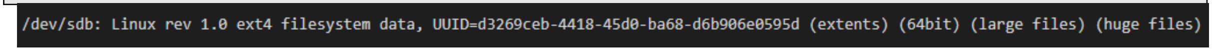


```
sudo mkfs -t ext4 /dev/sdb
mke2fs 1.44.1 (24-Mar-2018)
Creating filesystem with 2621440 4k blocks and 655360 inodes
Filesystem UUID: d3269ceb-4418-45d0-ba68-d6b906e0595d
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

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)
```

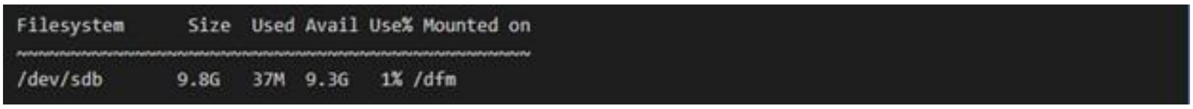
3) Mount "/dev/sdb" on /dfm

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

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

4) Verify

```
df -h
```



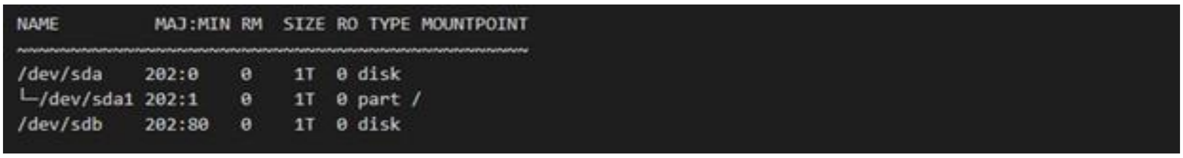
```
Filesystem      Size  Used Avail Use% Mounted on
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
/dev/sdb        9.8G   37M  9.3G   1% /dfm
```

【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**.

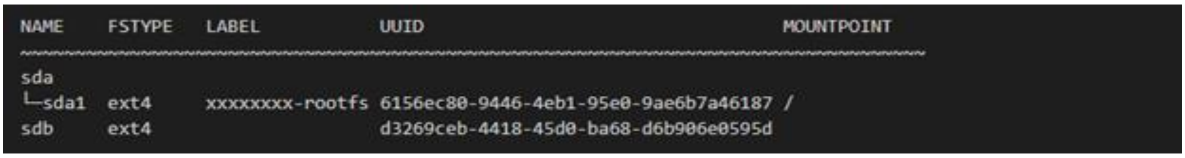
Now, let's check the disk information:

```
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

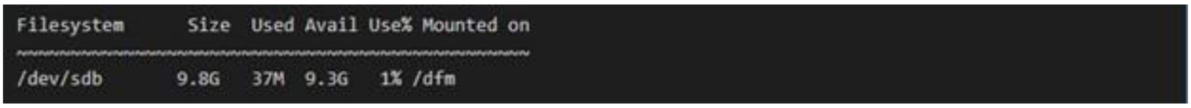
1) Mount /dev/sdb on /dfm

```
// 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
```

4.1.1 Permanently mount the disk

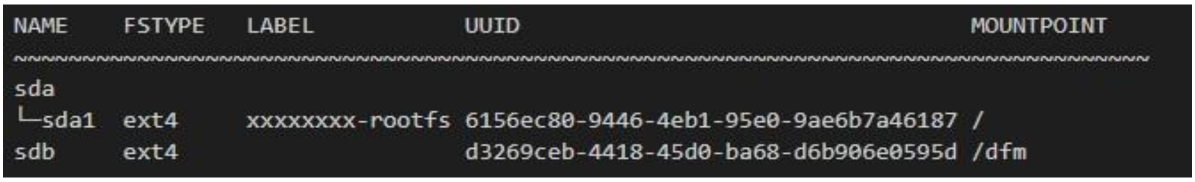
We recommend that the **customer's IT manager** sets the boot script so that **the dedicated disk** is auto-mounted when the server is booted.

If the **customer's IT manager** has not set the boot script for disk auto-mounting, you should proceed according to the command below.

*) If the settings are incorrect, Booting may not be possible. Also the command below is a general situation, and option may differ depending on the customer system situation. Please refer to the "fstab" menual for details.

1) Check mount /dev/sdb on /dfm

```
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
```

2) Edit /etc/fstab file

Please add the contents corresponding to "sdb" to the new line.

```
vi /etc/fstab

~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~
UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d    /dfm    ext4   defaults       0 0
```

4.2. Using Auto Install command

The auto-install package is delivered as a tar compress file.
This package contains the following resources:

- config_haproxy.sh	: Script for haproxy config
- dfm_config.conf	: dfm configuration file
- env.sh	: Script for environment config
- install.sh	: Script for installation
- run.sh	: Script for running install

4.2.1 Arguments

This step describes the parameters of the auto-install script.
The full execution instructions are as follows, and a detailed description of each parameter is included below.

```
./run.sh {argument1} {argument2} {argument3} {argument4} {argument5} {argument6} {argument7}
```

```
Example)
./run.sh install ubuntu nightwatch sec-dfm_1.0.1.11.deb dfm_config.conf /dev/sdb https example-sec-fota.net.pem
```

1) Argument 1

The arguments for installation commands.
Specified as “install” or “help”, the value is set to “install” during installation.

2) Argument 2

The arguments for the OS type.
“Ubuntu”, “redhat_1”, “redhat_2”, and “redhat_3” can be set to “redhat_1”, “redhat_2”, and “Ubuntu” in the current document.

3) Argument 3

The arguments for the name of the account to be installed.

The DFM Module is logged in with **a dedicated service account** and operates with the privileges of the account. Therefore, the dedicated service account has to be created in the server. The service account also needs the “**sudo**” privilege as a Docker requirement for command permissions. Ensure you add your service account into the Docker group.

If you want to use a service account, the following command shows you how to

add your service account into the Docker group.

We assume that you are using the “**nightwatch**” account.

```
$ sudo usermod -aG docker {your-user}

Example)
sudo usermod -aG docker nightwatch
```

4) Argument 4

The arguments for the name of the installation package file.
The DFM Module will either be delivered as a debian package or an rpm package tool.

The package naming rules are as follows.

```
sec-dfm_{version}.deb

Example)
sec-dfm_1.0.1.11.deb
```

5) Argument 5

The argument for the “dfm_config.conf” file
The “dfm_config.conf” file is provided as a compressed file and includes the information below.

【Configuration List】

```
[dfm_config]
host_ip=Static IP for DFM server.
listen_port= External listen port at server for DFM module to be accessed.
```

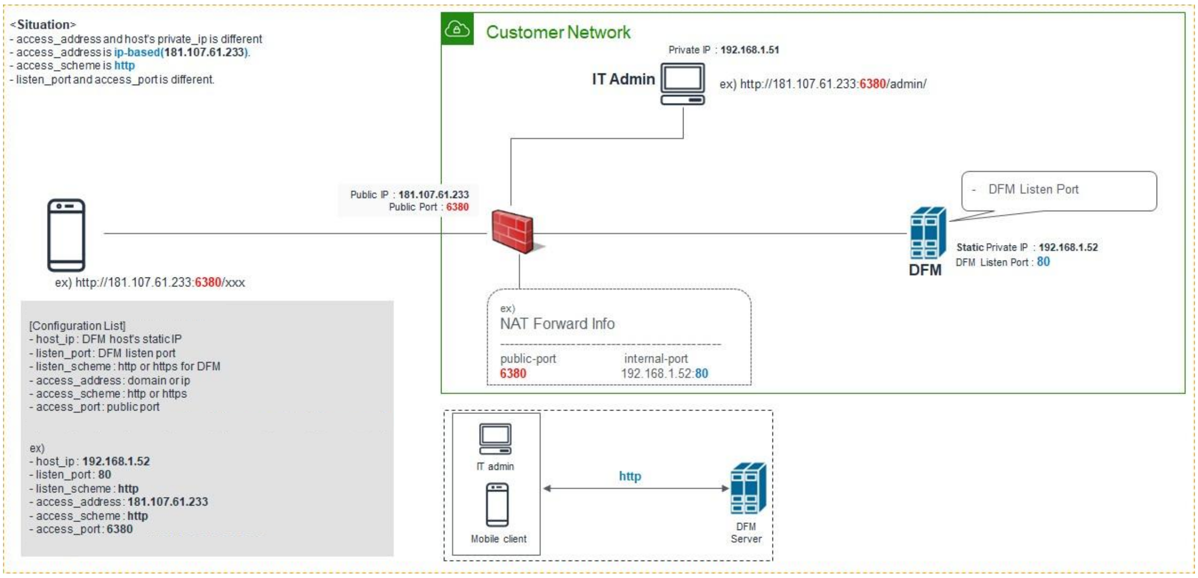
```
listen_scheme= url scheme(http or https) for DFM module to be accessed.
access_address= domain-based or ip-based
access_scheme= http or https
access_port= public port
```

In order to properly configure this service before 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.



<Situation>
- access_address and host's private_ip is different
- access_address is ip-based(181.107.61.233).
- access_scheme is http
- listen_port and access_port is different.

ex) http://181.107.61.233:6380/xxx

[Configuration List]
- host_ip: DFM host's static IP
- listen_port: DFM listen port
- listen_scheme: http or https for DFM
- access_address: domain or ip
- access_scheme: http or https
- access_port: public port

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

ex) http://181.107.61.233:6380/admin/

ex)
NAT Forward Info

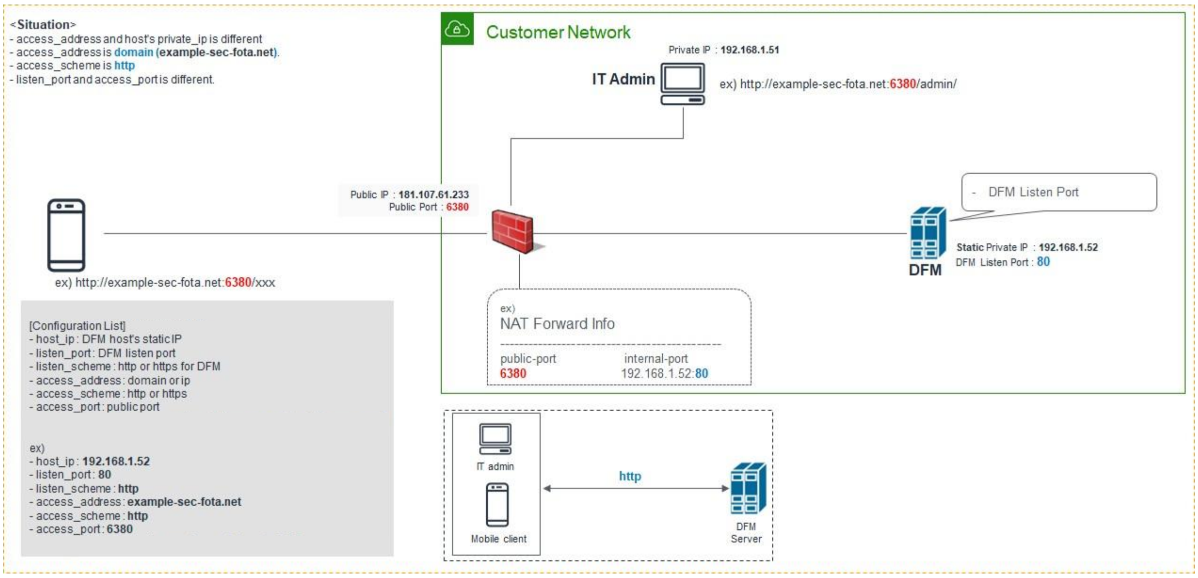
public-port	internal-port
6380	192.168.1.52:80

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**



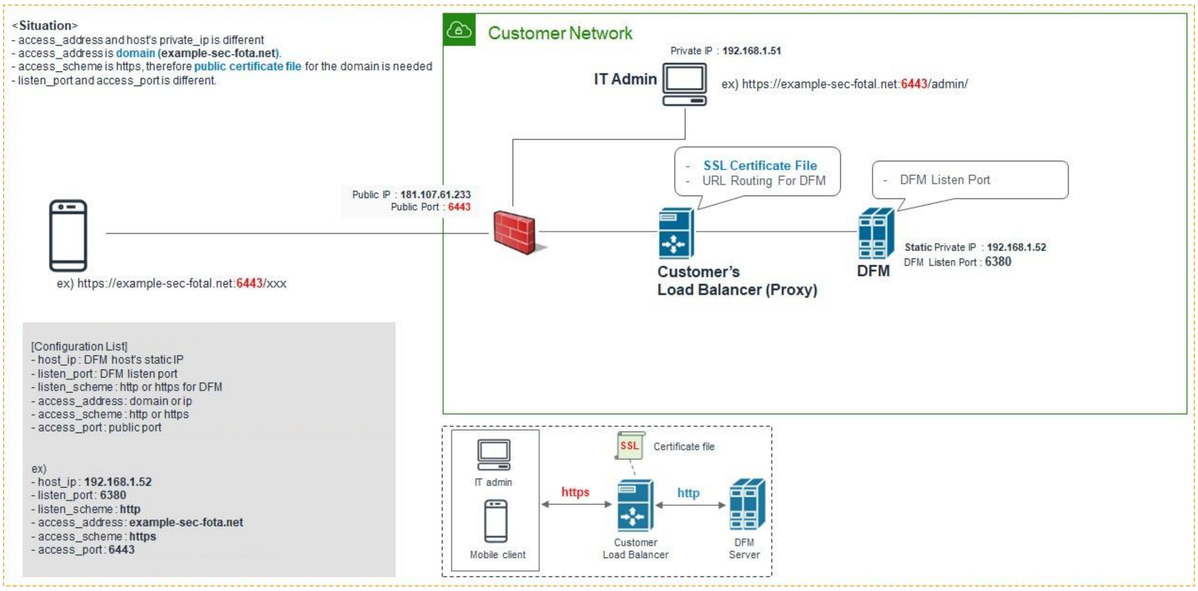
<Situation>
- access_address and host's private_ip is different
- access_address is domain (**example-sec-fota.net**).
- access_scheme is http
- listen_port and access_port is different.

[Configuration List]
- host_ip : DFM host's static IP
- listen_port : DFM listen port
- listen_scheme : http or https for DFM
- access_address : domain or ip
- access_scheme : http or https
- access_port : public port

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

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

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



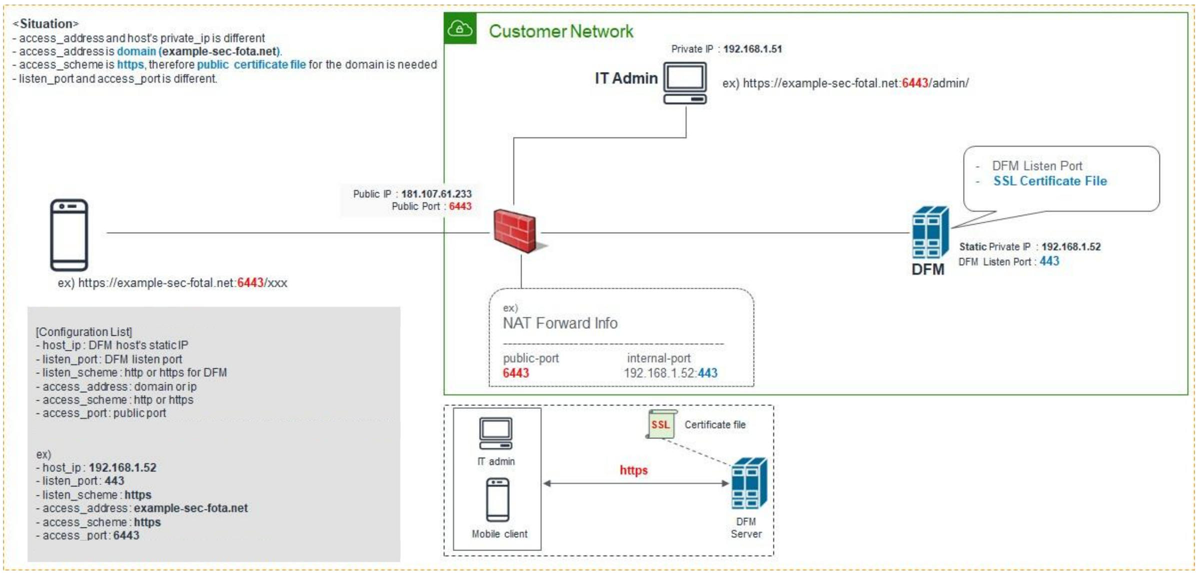
<Situation>
- access_address and host's private_ip is different
- access_address is domain (**example-sec-fota.net**).
- access_scheme is https, therefore public certificate file for the domain is needed
- listen_port and access_port is different.

[Configuration List]
- host_ip : DFM host's static IP
- listen_port : DFM listen port
- listen_scheme : http or https for DFM
- access_address : domain or ip
- access_scheme : http or https
- access_port : public port

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

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

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



<Situation>
- access_address and host's private_ip is different
- access_address is domain (example-sec-fota.net).
- access_scheme is https, therefore public certificate file for the domain is needed
- listen_port and access_port is different.

[Configuration List]
- host_ip : DFM host's static IP
- listen_port : DFM listen port
- listen_scheme : http or https for DFM
- access_address : domain or ip
- access_scheme : http or https
- access_port : public port

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

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

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

2.3.1 (CASE01) IP-Based

```
[dfm_config]
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:

```
vi dfm_config.conf

[dfm_config]
host_ip=192.168.1.52
listen_port=80
listen_scheme=http
access_address=181.107.61.233
access_scheme=http
access_port=6380
```

2.3.2 (CASE02) Domain-Based

⇦ (Type ①) Using **HTTP**

```
[dfm_config]
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:

```
vi dfm_config.conf

[dfm_config]
host_ip=192.168.1.52
listen_port=80
listen_scheme=http
access_address=example-sec-fota.net
access_scheme=http
access_port=6380
```

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

```
[dfm_config]
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:

```
vi dfm_config.conf

[dfm_config]
host_ip=192.168.1.52
listen_port=6380
listen_scheme=http
access_address=example-sec-fota.net
access_scheme=https
access_port=6443
```

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

```
[dfm_config]
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:

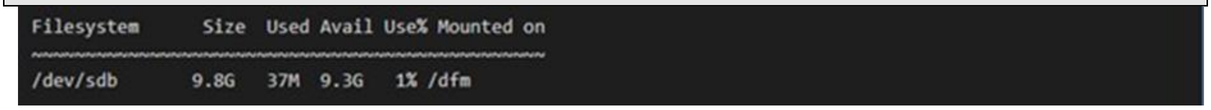
```
vi dfm_config.conf

[dfm_config]
host_ip=192.168.1.52
listen_port=443
listen_scheme=https
access_address=example-sec-fota.net
access_scheme=https
access_port=6443
```

6) Argument 6

The argument for mounting the disk.
Enter the file system name, checked using the command below, and ensure the installation folder is /dfm.

```
df –h
```



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

7) Argument 7

The argument for the proxy type.
The values that can be entered are “http”, “https”, and “proxy”.

If the external connection type is “https”, the customer must prepare the access domain they were issued and 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**” in “***Argument 5***”, this step must be completed.

I. HTTPS Handling

There are two possibilities for TLS/SSL Termination:

1) On Customer’s Load Balancer (Proxy)

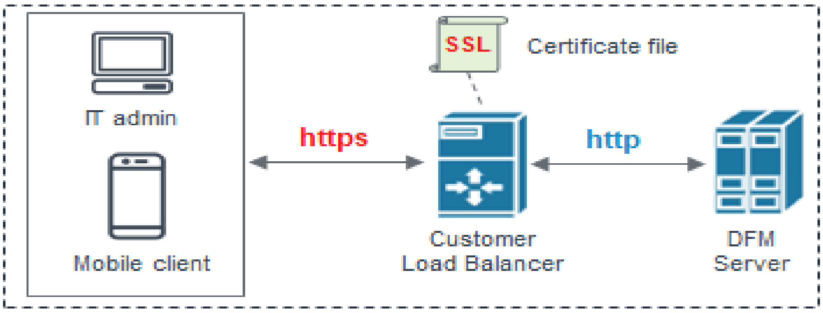


Fig 4-6 On Customer’s Load Balancer (Proxy)

In this case, we set the argument6 value to “proxy” and the customer’s IT manager will operate “public certificate” on its own Load Balancer.

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

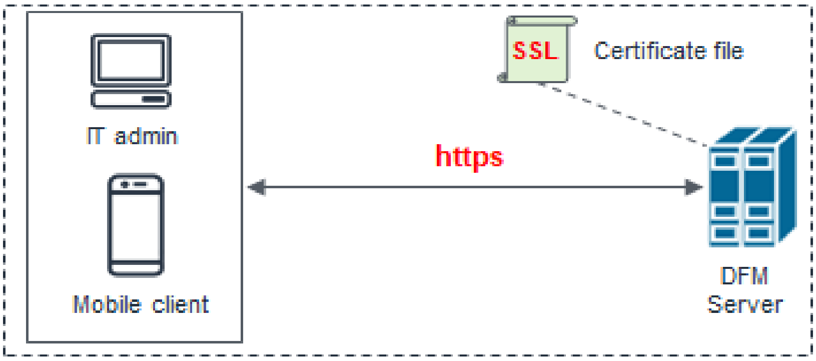


Fig 4-7 On DFM Server

In this case, we set Argument 6 to "https" and 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 auto install decompression package folder.

- 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 **example-sec-fota.net**

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

  ⇨ sudo -E bash -c 'cat fullchain.pem privkey.pem > example-sec-fota.net.pem'
  ⇨ '**example-sec-fota.net.pem**' is the unified certificate file

- Also you can make a pem file for devices. Copy the "chain.pem" file to create a new file named "efota.pem".
  ⇨ cp chain.pem efota.pem

II. HTTP Handling

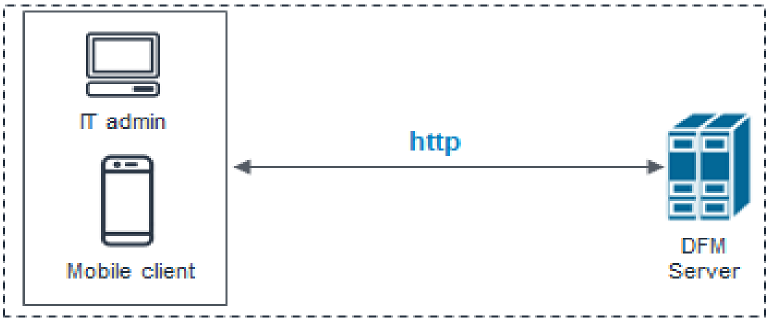


Fig 4-8 On DFM Server

In this case, we set the Argument 6 value to “http”.

8) Argument 8

The argument for the public certificate file.
This step only applies when the value of Argument 7 is set to “https”.

If the customer sets the Argument 7 value to “https”, 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 the auto install decompression package folder.

- 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 **example-sec-fota.net**

  ⇨ cert.pem

  ⇨ chain.pem

  ⇨ fullchain.pem: cert.pem and chain.pem combined

  ⇨ privkey.pem
  ⇨ sudo -E bash -c 'cat fullchain.pem privkey.pem > example-sec-fota.net.pem'
  ⇨ ‘**example-sec-fota.net.pem**’ is the unified certificate file

4.2.2 How to run Install Script

If the definition for each argument is complete, refer to the description below and execute it.
The argument used in the example below is as follows.

```
argument 1: install
argument 2: redhat_2
argument 3: nightwatch
argument 4: sec-dfm_1.0.1.11-root-enforcing.tar.gz
argument 5: dfm_config.conf
argument 6: /dev/sdb
argument 7: https
argument 8: example-sec-fota.net.pem
```

【**STEP01**】Extract the package.

```
//extract package
$ tar -zxvf OnPrem.tar.gz -C ~/

//check file
$ ls -l ~/OnPrem
```

```
-rwxrwxr-x. 1 2770 Dec  2 05:47 config_haproxy.sh
-rw-rw-r--. 1 134 Dec  2 05:47 dfm_config.conf
-rwxrwxr-x. 1 7907 Dec  2 05:47 env.sh
-rwxrwxr-x. 1 28187 Dec  2 05:47 install.sh
-rwxrwxr-x. 1 3500 Dec  2 05:47 run.sh
```

【STEP02】 Copy the DFM module package.
Move the Argument 4 file to the package folder.

```
//move dfm package
$ cp sec-dfm_1.0.1.11.deb ~/OnPrem

//check file
$ ls -l ~/OnPrem

-rwxrwxr-x. 1 2770 Dec  2 05:47 config_haproxy.sh
-rw-rw-r--. 1 134 Dec  2 05:47 dfm_config.conf
-rwxrwxr-x. 1 7907 Dec  2 05:47 env.sh
-rwxrwxr-x. 1 28187 Dec  2 05:47 install.sh
-rwxrwxr-x. 1 3500 Dec  2 05:47 run.sh
-rw-rw-r--. 1 516172613 May 26  2022 sec-dfm_1.0.1.11.deb
```

【STEP03】 Update the “dfm_config.conf” file.
Refer to (**[Use Case 2]:Type C**) in “*5) Argument 5*”

```
//edit dfm_config.conf
$ vi ~/OnPrem/dfm_config.conf

[dfm_config]
host_ip=192.168.1.52
listen_port=443
listen_scheme=https
access_address=example-sec-fota.net
access_scheme=https
access_port=6443
```

【Optional】 Copy the certificate file
This step applies only when a certificate is installed on the DFM server and assumes the value of Argument 7 is “https”.
Refer to (**[Use Case 2]:Type C**) in “*5) Argument 5*”

```
//copy certificate file
$ cp example-sec-fota.net.pem ~/OnPrem

//check file
$ ls -l ~/OnPrem
```

```
-rwxrwxr-x. 1 2770 Dec  2 05:47 config_haproxy.sh
-rw-rw-r--. 1 134 Dec  2 05:47 dfm_config.conf
-rwxrwxr-x. 1 7907 Dec  2 05:47 env.sh
-rwxrwxr-x. 1 28187 Dec  2 05:47 install.sh
-rwxrwxr-x. 1 3500 Dec  2 05:47 run.sh
-rw-rw-r--. 1 516172613 May 26  2022 sec-dfm_1.0.1.11.deb
-rw-r--r-- 1 7345 May 26  2022 example-sec-fota.net.pem
```

【STEP04】 Run Script

```
//copy certificate file
$ ./run.sh install ubuntu nightwatch sec-dfm_1.0.1.11.deb dfm_config.conf /dev/sdb https
example-sec-fota.net.pem
```

4.3. Change database password

We have to change the root account password after the installation is complete.

In this stage, you will perform the following:

1) Set the DB root password
2) Check the changed password

To do this, you must service the mysql server container. The command to check the mysql server container is as follows:

Make sure the MySQL container is in a healthy state.

```
$ docker ps –a | grep dfm-mysql

CONTAINER ID  IMAGE                                  STATUS                        NAMES
2cd1bae13406  localhost/mysql/enterprise-server:8.0  Up 52 seconds ago (healthy)   dfm-mysql
```

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}'

```
docker 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) Check the changed password

```
docker exec -it dfm-mysql mysql -uroot –p1q2w3e4r
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>
```

4.4. How to check Server Operation Status

Finally, the installer has completed the installation of the on-premises service-based Docker, 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 shown below. If every status returns healthy, the service is ready for operation.

```
docker ps -a


Example)
docker ps -a
  CONTAINER ID       ~      STATUS                              ~        NAMES
07ffa549f3cf             Up 2 minutes (healthy)               dfm-console
a470bb8bb995             Up 5 minutes (healthy)               dfm-core
af3949b8db98             Up 6 minutes (healthy)               dfm-minio
d882c61ba91c             Up 15 hours (healthy)                dfm-mysql
e10be66fe8bc             Up 3 minutes (healthy)               dfm-proxy
```

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 the admin console page after installation

If you completed installation, access the admin page to check whetherthe DFM Service is successfully installed and working.

【URL to the admin site】

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

⇒ Refer to "***5) Argument 5***".

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

```
[dfm_config]
host_ip=192.168.1.52
listen_port=80
listen_scheme=http
access_address=181.107.61.233
access_scheme=http
access_port=6380
```

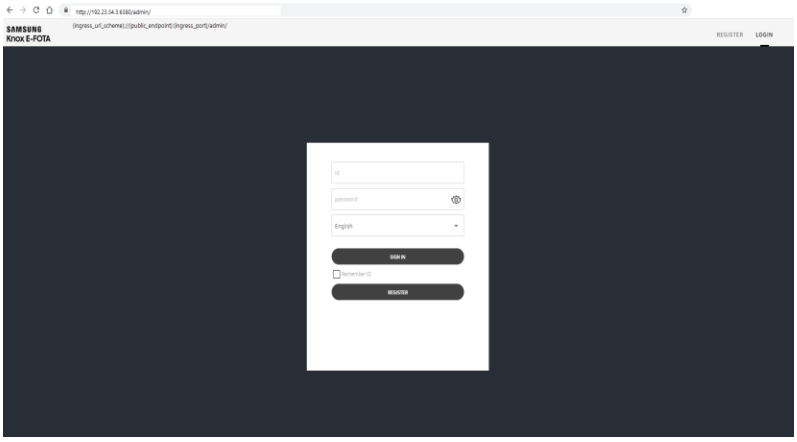
【Account & Initial Password (PWD)】

⇒ 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://181.107.61.233:6380/admin/ (using a new **Chrome** browser)



http://192.23.34.3:6380/admin/

{ingress_url_scheme}://{public_endpoint}:{ingress_port}/admin/

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 "***5) Argument 5***"

2) On DFM Server
   We need to update the certificate file on the DFM Server.
   Refer to (**[Use Case 2]:Type C**) in "***5) Argument 5***"

We assume that the newly 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 newly 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 its state is healthy.

```
  docker ps -a

  docker ps -a
CONTAINER ID     ~    STATUS                              ~                  NAMES
~~
e10be66fe8bc          Up 18 seconds (health: starting)                      dfm-proxy
~~
$

  docker ps -a
CONTAINER ID     ~    STATUS                              ~    NAMES
~~
e10be66fe8bc          Up 3 minutes (healthy)                   dfm-proxy
~~
$
```

5.5. DB Backup & Restore.

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

5.5.1. Backup a MySQL Server Instance

I. *To BACKUP a MySQL Server instance running in a Docker container using MySQL Shell Backup with Docker:*

1. On the same host where the MySQL Server container is running, start another container with an image of MySQL to perform a backup with the MySQL Shell Backup. 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 docker image currently installed is mysql:8.0.36.

【Basic Command】

```
docker run --rm --network container:dfm-mysql -v =/path-on-host-machine/datadir/:/backups mysql:8.0.36 mysqlsh --no-defaults -u{user} -p{password} -e "util.dumpInstance('/backups', {threads: 16, compression: 'zstd'})"
```

【Example】

```
sudo docker run --rm --network container:dfm-mysql -v /dfm/mysql/backups:/backups mysql:8.0.36
mysqlsh --no-defaults -uroot -p1q2w3e4r -e "util.dumpInstance('/backups', {threads: 16, compression:
'zstd'})"

nightwatch@xx:~$ docker run --rm --network container:dfm-mysql -v /dfm/mysql/backups:/backups mysql:8.0.36 mysqlsh
--no-defaults -uroot -p1q2w3e4r -e "util.dumpInstance('/backups', {threads: 16, compression: 'zstd'})"
Cannot set LC_ALL to locale en_US.UTF-8: No such file or directory
WARNING: Using a password on the command line interface can be insecure.
Initializing...
Acquiring global read lock
Global read lock acquired
Initializing - done
Gathering information...
2 out of 6 schemas will be dumped and within them 75 tables, 0 views.
3 out of 6 users will be dumped.
Gathering information - done
All transactions have been started
Locking instance for backup
Global read lock has been released
Writing global DDL files
Writing users DDL
Writing schema metadata...
Writing DDL...
Writing table metadata...
Running data dump using 16 threads.
Dumping data...
NOTE: Table statistics not available for `dfm_console_db`.`tb_system`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_console_db`.`tb_system`;' first.
....
```

```
NOTE: Table statistics not available for `dfm_console_db`.`tb_workspace_user`, chunking operation may be not
optimal. Please consider running 'ANALYZE TABLE `dfm_console_db`.`tb_workspace_user`;' first.
NOTE: Table statistics not available for `dfm_console_db`.`tb_domain`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_console_db`.`tb_domain`;' first.
NOTE: Table statistics not available for `dfm_console_db`.`tb_content_resource`, chunking operation may be not
optimal. Please consider running 'ANALYZE TABLE `dfm_console_db`.`tb_content_resource`;' first.
NOTE: Table statistics not available for `dfm_console_db`.`databasechangeloglock`, chunking operation may be not
optimal. Please consider running 'ANALYZE TABLE `dfm_console_db`.`databasechangeloglock`;' first.
NOTE: Table statistics not available for `dfm_core_db`.`tczs_lcs`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_core_db`.`tczs_lcs`;' first.
NOTE: Table statistics not available for `dfm_console_db`.`tb_group`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_console_db`.`tb_group`;' first.
NOTE: Table statistics not available for `dfm_core_db`.`tczs_dmn`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_core_db`.`tczs_dmn`;' first.
NOTE: Table statistics not available for `dfm_core_db`.`tczs_agent_pkg`, chunking operation may be not optimal.
Please consider running 'ANALYZE TABLE `dfm_core_db`.`tczs_agent_pkg`;' first.
NOTE: Table statistics not available for `dfm_core_db`.`tczs_fw_scnr`, chunking operation may be not optimal. Please
consider running 'ANALYZE TABLE `dfm_core_db`.`tczs_fw_scnr`;' first.
Writing schema metadata - done
Writing DDL - done
Writing table metadata - done
Starting data dump
Dumping data - done
Dump duration: 00:00:00s
Total duration: 00:00:00s
Schemas dumped: 2
Tables dumped: 75
Uncompressed data size: 1.18 MB
Compressed data size: 288.85 KB
Compression ratio: 4.1
Rows written: 1968
Bytes written: 288.85 KB
Average uncompressed throughput: 1.18 MB/s
Average compressed throughput: 288.85 KB/s
```

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/
```

5.5.2. Restore a MySQL Server Instance

II. *To RESTORE a MySQL Server instance in a Docker container using MySQL Shell with Docker:*

1. Stop the MySQL Server container, which also stops the MySQL Server running inside:

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. Restart the server container

```
dfm start dfm-mysql
```

4. Set DB root password(we assume that “password” is **“1q2w3e4”**) and and set the ‘local_infile’ for restore

```
docker exec -it dfm-mysql mysql -uroot -e "ALTER USER 'root'@'localhost' IDENTIFIED BY '1q2w3e4r';"
docker exec -it dfm-mysql mysql -uroot -p1q2w3e4r -e "set global local_infile = 1;"
```

5. Start a container with an image of MySQL to perform the restore with the MySQL Shell. 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】

```
docker run --rm --network container:dfm-mysql -v /dfm/mysql/backups:/backups mysql:8.0.36 mysqlsh --no-defaults -u{user} -p{password} -e "util.loadDump('/backups', {threads: 16})"
```

【Example】

```
docker run --rm --network container:dfm-mysql -v /dfm/mysql/backups:/backups mysql:8.0.36 mysqlsh --
no-defaults -uroot -p1q2w3e4r -e "util.loadDump('/backups', {threads: 16})"

nightwatch@xx~$ docker run --rm --network container:dfm-mysql -v /dfm/mysql/backups:/backups
mysql:8.0.36 mysqlsh --no-defaults -uroot -p1q2w3e4r -e "util.loadDump('/backups', {threads: 16})"
Cannot set LC_ALL to locale en_US.UTF-8: No such file or directory
WARNING: Using a password on the command line interface can be insecure.
Loading DDL and Data from '/backups' using 16 threads.
Opening dump...
Target is MySQL 8.0.36. Dump was produced from MySQL 8.0.36
Scanning metadata...
Scanning metadata - done
Checking for pre-existing objects...
Executing common preamble SQL
Executing DDL...
Executing DDL - done
Executing view DDL...
Executing view DDL - done
Loading data...
Starting data load
Recreating indexes...
Loading data - done
Recreating indexes - done
Executing common postamble SQL
75 chunks (1.97K rows, 1.18 MB) for 75 tables in 2 schemas were loaded in 2 sec (avg throughput 1.18
MB/s)
0 warnings were reported during the load.
```

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

6. Off the ‘local_infile’ option

```
docker exec -it dfm-mysql mysql -uroot -p1q2w3e4r -e "set global local_infile = 0;"
```

5.6 Configurable length of password digits

The installer can change this default value of a minimum and maximum length of password digits. (default password_min_length=8, default password_max_length=12)

【STEP01】 Stop DFM Admin Console

The command to stop the DFM Admin Console Server container is as follows

```
dfm terminate dfm-console
```

【STEP02】 Set-up the length of the password digits

The minimum length of password is allowed from 8 to 20.
The max length of password is allowed from 12 to 30.

```
dfm config set password_min_length=8
dfm config set password_min_length=20
```

【STEP03】Check the length of the password digits

```
dfm config get password_min_length
8

dfm config get password_max_length
20
```

【STEP04】Restart DFM Admin Console

The command to restart the DFM Admin Console Server container is as follows

```
dfm start dfm-console
```

To make sure that the DFM Admin Console container is in a healthy state, run the following command. It maytake some time until its state is healthy.

```
    docker ps -a

    docker ps -a
   CONTAINER ID   ~    STATUS                              ~                NAMES

~~ e92be16ye8bc        Up 18 seconds (health: starting)                   dfm-console
~~
$

   docker ps -a
  CONTAINER ID     ~    STATUS                              ~    NAMES

~~ e92be16ye8bc         Up 3 minutes (healthy)                    dfm-console
~~
$
```

5.7 Configurable device group polling

The installer can change the default value of the device group. (default device_group_enable=false, device_group_max_limit=20000)
This function is used to distribute a large number of devices when serving, and all the devices are distributed across 60 groups.

【STEP01】Stop DFM Core

The command to stop the DFM Core Server container is as follows:

```
dfm terminate dfm-core
```

【STEP02】Set up the device group polling

The allowed values for “device group enable” are “true” or “false”.
The device group max limit is 20000.

```
dfm config set device_group_enable =true
dfm config set device_group_max_limit =20000
```

【STEP03】 Check the device group polling

```
dfm config get device_group_enable
true

dfm config get device_group_max_limit
20000
```

【STEP04】 Restart the DFM Core

The command to restart the DFM Core Server container is as follows:

```
dfm start dfm-core
```

To make sure that the DFM Admin Console container is in a healthy state, run the following command. It may take some time until its state is healthy.

```
    docker ps -a

    docker ps -a
   CONTAINER ID  ~    STATUS                              ~                 NAMES

~~ e92be16ye8bc       Up 18 seconds (health: starting)                      dfm-core
~~
$

   docker ps -a
  CONTAINER ID   ~    STATUS                              ~    NAMES

~~ e92be16ye8bc       Up 3 minutes (healthy)                   dfm-core
~~
$
```

5.8 Configurable device polling interval and postpone waiting time

The installer can change the default value of the device polling interval and postpone waiting time. (default polling_interval_register=86400, default_waiting_time=30)

【STEP01】Stop DFM Core

The command to stop the DFM Core Server container is:

```
dfm terminate dfm-core
```

【STEP02】Set up the device polling interval and postpone waiting time

The polling_interval_register can only be an integer.
The postpone waiting time limit is allowed from 1 to 7200.

```
dfm config set polling_interval_register =86400
dfm config set default_waiting_time =30
```

【STEP03】Check the device polling interval and postpone waiting time

```
dfm config get polling_interval_register
86400

dfm config get default_waiting_time
30
```

【STEP04】Restart DFM Core

The command to restart the DFM Core Server container is:

```
dfm start dfm-core
```

To make sure that the DFM Admin Console container is in a healthy state, run the following command. It maytake some time until its state is healthy.

```
    docker ps -a

    docker ps -a
   CONTAINER ID  ~   STATUS                          ~                NAMES

~~ e92be16ye8bc      Up 18 seconds (health: starting)                 dfm-core
~~
$


                 ~   STATUS                          ~   NAMES

                     Up 3 minutes (healthy)              dfm-core

```

6. When a Server is Rebooted

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:

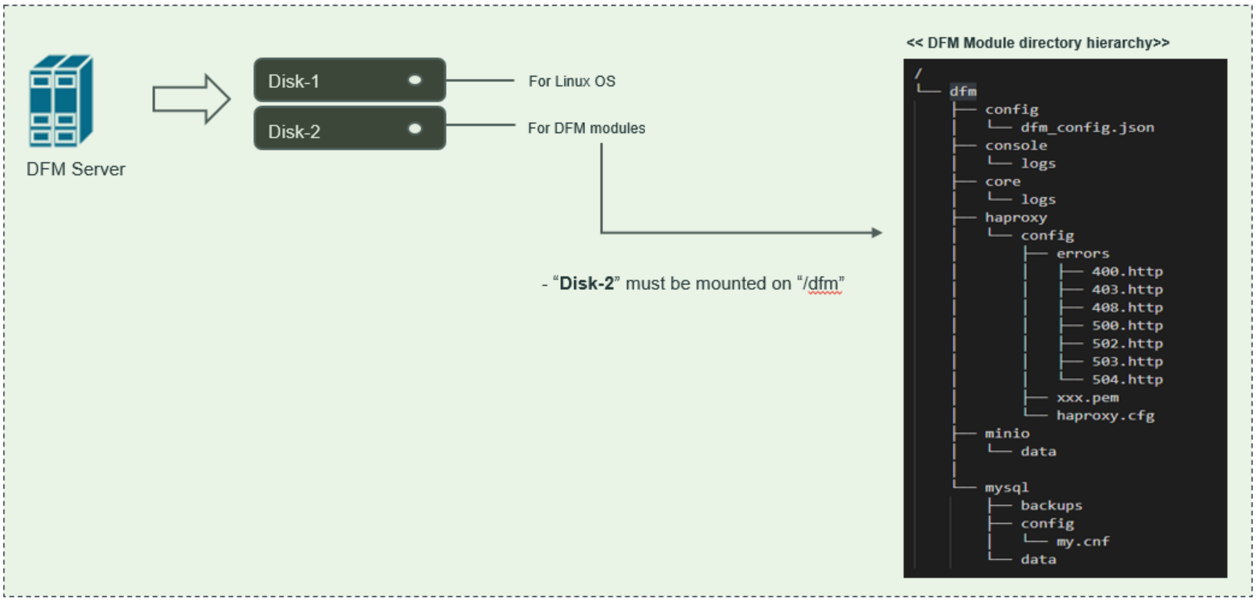
6.1. (STEP01) Login as the dedicated service account

The DFM Module is logged in with a dedicated service account and operates with the privileges of the account (see, "***3) Argument 3***").

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**.


DFM Server
Disk-1
Disk-2
For Linux OS
For DFM modules
- "**Disk-2**" must be mounted on "/dfm"
<< DFM Module directory hierarchy>>

```
/
└── dfm
    ├── config
    │   └── dfm_config.json
    ├── console
    │   └── logs
    ├── core
    │   └── logs
    ├── haproxy
    │   └── config
    │       ├── errors
    │       │   ├── 400.http
    │       │   ├── 403.http
    │       │   ├── 408.http
    │       │   ├── 500.http
    │       │   ├── 502.http
    │       │   ├── 503.http
    │       │   └── 504.http
    │       ├── xxx.pem
    │       └── haproxy.cfg
    ├── minio
    │   └── data
    │
    └── mysql
        ├── backups
        ├── config
        │   └── my.cnf
        └── data
```

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

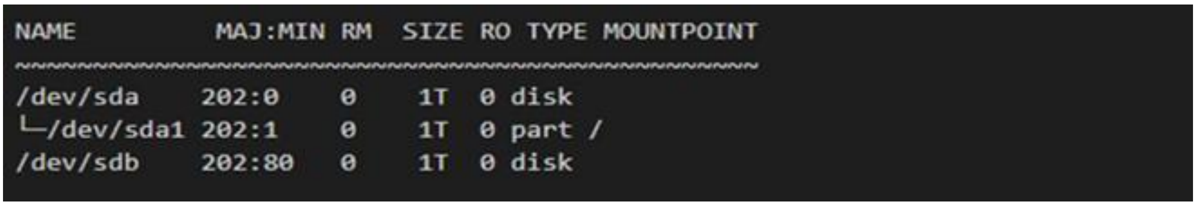
For example, we assume that two disks ("sda" and "sdb") exist.

【CASE01】 Disk is Ready

If the disk is ready, we don't need to mount it.

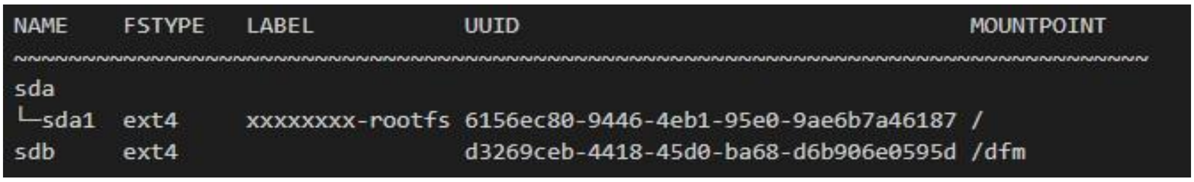
Now, let's check the disk information:

```
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
```

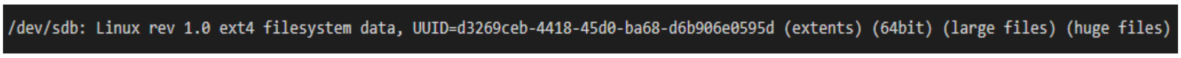
```
sudo lsblk –f
```



```
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
```

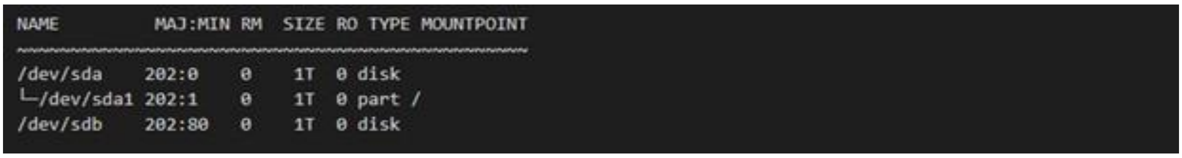


```
/dev/sdb: Linux rev 1.0 ext4 filesystem data, UUID=d3269ceb-4418-45d0-ba68-d6b906e0595d (extents) (64bit) (large files) (huge files)
```

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

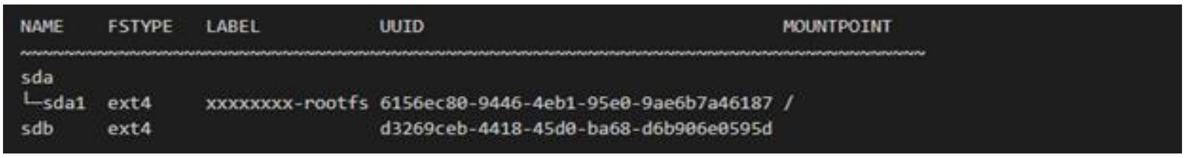
If the disk is formatted but not yet mounted, it needs to be mounted on **/dfm**.
Now, let’s check the disk information.

```
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

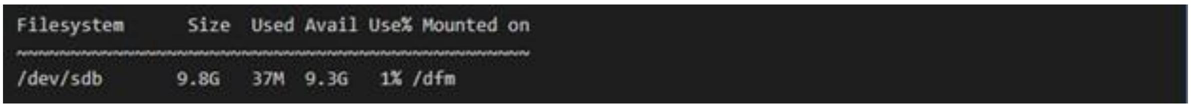
1) Mount /dev/sdb on /dfm

```
// 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 Docker

After the system is rebooted, check whether the Docker engine is running.

```
  sudo systemctl status docker
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
   docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Drop-In: /etc/systemd/system/docker.service.d
                    └─http-proxy.conf, https-proxy.conf
   Active: active (running) since Fri 2020-02-07 13:12:39 KST; 3 weeks 2 days ago
     Docs: https://docs.docker.com
~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
```

If the **Active** value is not "**active (running)**", Docker is not yet running.

If the Docker engine is not running, run it using the following command.

```
$ sudo systemctl start docker
```

6.4. (STEP04) 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.

```
  docker ps -a

Example)
$ docker ps -a
CONTAINER ID      ~     STATUS                              ~        NAMES
~~
d882c61ba91c            Up 4 seconds (health: starting)              dfm-mysql
~~

  docker ps -a
CONTAINER ID      ~     STATUS                              ~     NAMES
~~
d882c61ba91c            Up 2 minutes (healthy)                     dfm-mysql
~~
```

6.5. (STEP05) Start up Firmware Storage Server

After the system is rebooted, restart Minio. The command to run 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.

```
  docker ps -a

Example)
docker ps -a
CONTAINER ID     ~    STATUS                              ~        NAMES
~~
af3949b8db98          Up 4 seconds (health: starting)             dfm-minio
~~

  docker ps -a
CONTAINER ID     ~    STATUS                              ~     NAMES
~~
af3949b8db98          Up 2 minutes (healthy)                     dfm-minio
~~
```

6.6. (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.

```
  docker ps -a

Example)
  docker ps -a
CONTAINER ID     ~    STATUS                              ~     NAMES
~~
a470bb8bb995           Up 4 seconds (health: starting)          dfm-core
~~
$

  docker ps -a
CONTAINER ID     ~    STATUS                              ~     NAMES
~~
a470bb8bb995           Up 2 minutes (healthy)                  dfm-core
```

```
~~
```

6.7. (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 takes some time until its state is healthy.

```
  docker ps -a

Example)
  docker ps -a
CONTAINER ID      ~     STATUS                              ~                 NAMES
~~
07ffa549f3cf          Up 7 seconds (health: starting)                        dfm-console
~~

  docker ps -a
CONTAINER ID      ~     STATUS                              ~                 NAMES
~~
07ffa549f3cf          Up 2 minutes (healthy)                                 dfm-console
~~
```

6.8. (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 make sure the HAProxy container is in a healthy state. It takes some time until its state is healthy.

```
  docker ps -a

Example)
  docker ps -a
CONTAINER ID      ~     STATUS                       ~                NAMES
~~
e10be66fe8bc            Up 18 seconds (health: starting)              dfm-proxy
~~
```

```
  docker ps -a
CONTAINER ID      ~     STATUS                          ~          NAMES
~~
e10be66fe8bc           Up 3 minutes (healthy)                      dfm-proxy
~~
```

6.9. (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.

```
  docker ps -a

Example)
  docker ps -a
CONTAINER ID      ~     STATUS                          ~     NAMES
e10be66fe8bc           Up 3 minutes (healthy)                 dfm-proxy
07ffa549f3cf           Up 14 minutes (healthy)                dfm-console
a470bb8bb995           Up 16 minutes (healthy)                dfm-core
af3949b8db98          Up 17 minutes (healthy)                dfm-minio
d882c61ba91c          Up 15 hours (healthy)                  dfm-mysql
```

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. Docker Image Update

If there is an updated DFM Module, it is released as a Docker Image Package and packed as a tar file. In the release, the Docker 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:

- docker 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 Docker Image.

```
docker load < dfm-mysql-xx.xx.xx.tar
```

【STEP03】 Change the repository and tag's configuration

```
dfm config set mysql_img_rep=dfm-mysql
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.

```
docker ps -a
```

- 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.

```
docker ps -a
```

- DFM Admin Console Server

```
dfm start dfm-console
```

【Validation】

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

```
docker ps -a
```

7.1.2. DFM Firmware Storage Update (MinIO)

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

- docker 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 Docker Image.

```
docker load < 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.

```
docker ps -a
```

7.1.3. DFM Core Update

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

- docker 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 docker image.

```
docker load < 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.

  ```
  docker ps -a
  ```

7.1.4. DFM Admin Console Update

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

- docker 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 docker image.

```
docker load < 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 ensure the mysql container is in a healthy state. It takes some time until its state is healthy.

```
docker ps -a
```

7.1.5. HAProxy update

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

- docker 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 docker image.

```
docker load < 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.

  ```
  docker 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. Purge the installed Debian package

【STEP01】 Check if the installed dfm debian package exists.

```
dpkg -l | grep sec-dfm

example-1) when installed pkg exists
$ dpkg -l | grep sec-dfm
ii    sec-dfm         1.0.1.11     all Samsung Enterprise fota dfm package
$

example-2) when installed pkg does Not exist
$ dpkg -l | grep sec-dfm
$
```

【STEP02】 If the installed dfm debian package exists, remove it.

```
sudo dpkg -P sec-dfm

example)
$ sudo dpkg -P sec-dfm
(Reading database ... 300748 files and directories currently installed.)
Removing sec-dfm (1.0.1.11) ...
$
```

8.2. Terminate Services

If there are active services, terminate them.

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

```
docker ps -a

example)
docker ps -a
CONTAINER ID   IMAGE                                        ~~ STATUS                    ~~ NAMES
879ab3603220   dfm-console:1.0.1.11                         ~~ Up 3 hours (healthy)   ~~   dfm-console
2966ea3ab692   dfm-core:1.0.1.11                            ~~ Up 3 hours (healthy)   ~~   dfm-core
b6ed98da1101   haproxytech/haproxy-debian:2.2.33            ~~ Up 3 hours (healthy)   ~~    dfm-proxy
b10b70f135d0   minio/minio:  RELEASE.2022-04-30T22-23-53Z   ~~ Up 3 hours (healthy)   ~~   dfm-minio
63c384eb0d5c   mysql/mysql-8.0.36                           ~~ Up 3 hours (healthy)   ~~   dfm-mysql
```

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
   ```

7. Stop Background App and check

   Stop background app with the following command:

   ```
   sudo systemctl stop efota-license.service
   ```

   Check with the following command:

   ```
   sudo systemctl status efota-license.service
     Loaded: loaded (/etc/system/system/efota-license.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since Tue 2024-XX-XX 06:39:10 UTC; 7s ago
   ```

8.3. 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-Premises
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 22.04.4 LTS 24.04 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	Community Edition	Community (Free)	
PoC	**Server OS**	Ubuntu	18.04.3 LTS 22.04.4 LTS 24.04 LTS	Community (free)	
	Container	Docker Engine	Community Edition	Community (Free)	
	Database	MySQL	Community Edition	Community (Free)	

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 “**Knox E-FOTA On-Premises 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 Installation 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 into the Server 4. Upload the Firmware deltas (Contents for FOTA) 5. Upload the device information using during the test 6. Create the Campaign 7. Perform 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-Premises.pdf**” 2. Introduce how to raise issues ☜ Using “**Issue raising process**” 3. Introduce service operation steps ☜ Using “**Service Operation Guide**” 4. Sign the “Notice for Completion Installation”

		☜ Refer to "**Appendix E**" (*Installation and Initial Operation Guide for Knox E-FOTA On-Premises.pdf*) 5. 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) Name:_____________________________ Signature:__________________________ Date: ______________________________	**< Customer Name >** Name:_____________________________ Signature:__________________________ 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.

Appendix F. Set E-FOTA agent config by managed Configuration

The Knox E-FOTA On-Premises agent requires your server URL and TLS certificate to connect to your on-premises server. The Knox E-FOTA On-Premises agent supports a managed configuration which lets you configure connection and authentication details needed for the app to communicate with your on-premises server. Follow your EMM's documentation on how to configure an app's managed configuration.

Reference: https://developer.android.com/work/managed-configurations

The managed configuration has the following fields:

Field	Description
server_url	*string* The URL of your Knox E-FOTA On-Premises server
pem	*string* Your server's TLS certificate
client_cert_pem	*string* (Only used for mutual TLS) Client certificate used by clients to authenticate themselves to your server
client_key_pem	*string* (Only used for mutual TLS) Encrypted private key used by clients for signing
client_pem_password	*string* (Only used for mutual TLS) Password for the encrypted private key

< **EOF** (**End Of File**) >