



NetIQ® Identity Manager Deploying Identity Manager 4.8.2 Containers

October 2020

Legal Notice

For information about NetIQ legal notices, disclaimers, warranties, export and other use restrictions, U.S. Government restricted rights, patent policy, and FIPS compliance, see <https://www.netiq.com/company/legal/>.

Copyright (C) 2020 NetIQ Corporation. All rights reserved.

Contents

About this Book and the Library	5
About NetIQ Corporation	7
Part I Overview and Planning of Container Deployment	9
1 Planning Your Container Deployment	11
System Requirements	11
Obtaining the Docker Images	11
2 Fresh Deployment of Identity Manager Containers	13
Preparing Your Container Deployment	13
Managing Container Volume Data	14
Prerequisites for Deploying Containers	14
Creating the Silent Properties File	16
Deploying Containers on a Single Server	17
Deploying Identity Manager Engine Container	18
Deploying Remote Loader Container	18
Deploying Fanout Agent Container	19
Deploying iManager Container	19
Generating Certificate With Identity Vault Certificate Authority	21
Deploying OSP Container	22
Deploying PostgreSQL Container	23
Deploying Identity Applications Container	24
Deploying Form Renderer Container	25
Deploying ActiveMQ Container	26
Deploying Identity Reporting Container	26
Deploying SSPR Container	27
Deploying Containers on Distributed Servers	28
Deploying Identity Manager Engine Container	30
Deploying Remote Loader Container	31
Deploying Fanout Agent Container	31
Deploying iManager Container	32
Generating Certificates With Identity Vault Certificate Authority	33
Deploying OSP Container	37
Deploying PostgreSQL Container	38
Deploying Identity Applications Container	39
Deploying Form Renderer Container	40
Deploying ActiveMQ Container	41
Deploying Identity Reporting Container	41
Deploying SSPR Container	42
3 Updating Identity Manager Containers	45
Prerequisites for Updating Containers	45
Updating Containers on a Single Server	45

Updating Identity Manager Engine Container	46
Updating Remote Loader Container	46
Updating Fanout Agent Container	47
Updating iManager Container	47
Updating OSP Container	48
Updating PostgreSQL Container	48
Updating Identity Applications Container	50
Updating Form Renderer Container	50
Updating ActiveMQ Container	50
Updating Identity Reporting Container	51
Updating SSPR Container	51
Updating Containers on Distributed Servers	51
Updating Identity Manager Engine Container	51
Updating Remote Loader Container	52
Updating Fanout Agent Container	53
Updating iManager Container	53
Updating OSP Container	54
Updating PostgreSQL Container	55
Updating Identity Applications Container	56
Updating Form Renderer Container	56
Updating ActiveMQ Container	57
Updating Identity Reporting Container	57
Updating SSPR Container	57
4 Troubleshooting	59
Identity Applications Container Displays Portlet Registration Exception	59
5 Best Practices	61

About this Book and the Library

This guide provides instructions for deploying the NetIQ Identity Manager (Identity Manager) containers.

Intended Audience

This book is intended for identity architects and identity administrators responsible for installing Identity Manager using containers.

Other Information in the Library

For more information about the library for Identity Manager, see the [Identity Manager documentation website](#).

About NetIQ Corporation

We are a global, enterprise software company, with a focus on the three persistent challenges in your environment: Change, complexity and risk—and how we can help you control them.

Our Viewpoint

Adapting to change and managing complexity and risk are nothing new

In fact, of all the challenges you face, these are perhaps the most prominent variables that deny you the control you need to securely measure, monitor, and manage your physical, virtual, and cloud computing environments.

Enabling critical business services, better and faster

We believe that providing as much control as possible to IT organizations is the only way to enable timelier and cost effective delivery of services. Persistent pressures like change and complexity will only continue to increase as organizations continue to change and the technologies needed to manage them become inherently more complex.

Our Philosophy

Selling intelligent solutions, not just software

In order to provide reliable control, we first make sure we understand the real-world scenarios in which IT organizations like yours operate—day in and day out. That's the only way we can develop practical, intelligent IT solutions that successfully yield proven, measurable results. And that's so much more rewarding than simply selling software.

Driving your success is our passion

We place your success at the heart of how we do business. From product inception to deployment, we understand that you need IT solutions that work well and integrate seamlessly with your existing investments; you need ongoing support and training post-deployment; and you need someone that is truly easy to work with—for a change. Ultimately, when you succeed, we all succeed.

Our Solutions

- ♦ Identity & Access Governance
- ♦ Access Management
- ♦ Security Management
- ♦ Systems & Application Management
- ♦ Workload Management
- ♦ Service Management

Contacting Sales Support

For questions about products, pricing, and capabilities, contact your local partner. If you cannot contact your partner, contact our Sales Support team.

Worldwide:	www.netiq.com/about_netiq/officelocations.asp
United States and Canada:	1-888-323-6768
Email:	info@netiq.com
Website:	www.netiq.com

Contacting Technical Support

For specific product issues, contact our Technical Support team.

Worldwide:	www.netiq.com/support/contactinfo.asp
North and South America:	1-713-418-5555
Europe, Middle East, and Africa:	+353 (0) 91-782 677
Email:	support@netiq.com
Website:	www.netiq.com/support

Contacting Documentation Support

Our goal is to provide documentation that meets your needs. The documentation for this product is available on the NetIQ website in HTML and PDF formats on a page that does not require you to log in. If you have suggestions for documentation improvements, click **comment on this topic** at the bottom of any page in the HTML version of the documentation posted at www.netiq.com/documentation. You can also email Documentation-Feedback@netiq.com. We value your input and look forward to hearing from you.

Contacting the Online User Community

NetIQ Communities, the NetIQ online community, is a collaborative network connecting you to your peers and NetIQ experts. By providing more immediate information, useful links to helpful resources, and access to NetIQ experts, NetIQ Communities helps ensure you are mastering the knowledge you need to realize the full potential of IT investments upon which you rely. For more information, visit <https://www.netiq.com/communities/>.

Overview and Planning of Container Deployment

Identity Manager provides the flexibility of deploying Identity Manager Components through a containerized mechanism. Identity Manager uses Docker for managing containers. The Identity Manager components, that support containerization, are delivered as Docker images. The Docker images are self-sufficient to run on their own.

All the functionalities and operations that can be achieved through the enterprise mode of installation are also available through the containerized mechanism.

However, the advantage of using containers is the ability to perform a fresh installation with every new version of containers along with the option of updating from previous versions. NetIQ recommends you to directly use the 4.8.2 version of containers if you are using containers for the first time.

1 Planning Your Container Deployment

The following sections describe the high-level planning required for a container-based deployment in Docker environment:

- ♦ “System Requirements” on page 11
- ♦ “Obtaining the Docker Images” on page 11

System Requirements

You must ensure that the following requirements are met for deploying the containers:

Software	Certified Versions
Docker	19.03.1 or later

Obtaining the Docker Images

Perform the following steps to obtain the Docker images:

- 1 Download the `Identity_Manager_4.8.2_Containers.tar.gz` from the [download page](#).
- 2 Run the following command to extract the `.tar` file:

```
tar -zxvf Identity_Manager_4.8.2_Containers.tar.gz
```


2 Fresh Deployment of Identity Manager Containers

This section guides you through the process of installing Identity Manager containers. After Identity Manager containers are deployed, you must perform some additional configuration steps for the components to be fully functional. For more information, see [Final Steps for Completing the Installation](#) section in the [NetIQ Identity Manager Setup Guide for Linux](#).

The Docker images are available for the following Identity Manager components:

- ♦ Identity Manager Engine
- ♦ Remote Loader
- ♦ iManager
- ♦ One SSO Provider (OSP)
- ♦ Fanout Agent
- ♦ ActiveMQ
- ♦ PostgreSQL (Redistribution)
- ♦ Identity Applications
- ♦ Self Service Password Reset (SSPR)
- ♦ Form Renderer
- ♦ Identity Reporting

NOTE: The Identity Configuration Generator image is used for generating the silent properties file. For information about creating the silent properties file, see [“Creating the Silent Properties File”](#) on page 16.

The procedures for deploying containers are described in subsequent sections.

- ♦ [“Preparing Your Container Deployment”](#) on page 13
- ♦ [“Deploying Containers on a Single Server”](#) on page 17
- ♦ [“Deploying Containers on Distributed Servers”](#) on page 28

Preparing Your Container Deployment

The Identity Manager containers deployment process requires pre-installation, installation, and post-installation work. Use the information in this section as you prepare to deploy the Identity Manager containers.

Some containers are dependent on others. The following table provides details on those containers that are dependent on other containers.

Table 2-1 *Dependent Containers*

Container	Dependent containers
OSP	<ul style="list-style-type: none">◆ Identity Engine◆ iManager
Identity Applications	<ul style="list-style-type: none">◆ OSP◆ Databases for Identity Applications
Form Renderer	Identity Applications
Identity Reporting	<ul style="list-style-type: none">◆ Identity Applications◆ Databases for Identity Reporting
SSPR	OSP

Managing Container Volume Data

Docker supports several mechanisms for data storage and persistence. One such mechanism of persisting container data is by using shared volumes in containers.

The examples used in this guide assumes that you create and use shared volumes. For example, create a shared volume called `/data` on your Docker host.

```
mkdir /data
```

However, you can use other volumes that Docker supports. For more information, see [Docker](#) documentation.

NOTE: The `/data` directory of the Docker host will be mapped to the `/config` directory of the containers. Ensure that you have read-write permissions for the shared volumes. However, if you want to map the shared volume with a different directory inside the container, you must map them while deploying the container itself. For example, you can map the `/data` directory with the `/etc/opt/novell/dirxml/rdxml/` directory inside the Remote Loader container.

Prerequisites for Deploying Containers

Based on your container deployment, NetIQ recommends that you review the following prerequisites before deploying containers.

- ◆ The `/etc/hosts` file of all the Docker hosts in your Docker deployment must be updated with the details of all the containers running on that host. Ensure that the hostname for all containers are in Fully Qualified Domain Name (FQDN) format only.
 - ◆ If you are deploying containers on a single server, ensure that the host file entry follows the below format:

```
<IP of the host> <FQDN> <short_name>
```

For example:

```
172.120.0.1      identitymanager.example.com      identitymanager
```

- ◆ If you are deploying containers on distributed servers, ensure that the host file entries follows the below format for all the components:

<IP of the container> <FQDN> <short_name>

In the sample deployment used in this guide, add the following entries in the /etc/hosts file:

```
192.168.0.12    identityengine.example.com    identityengine
192.168.0.2    remoteloader.example.com      remoteloader
192.168.0.3    fanoutagent.example.com       fanoutagent
192.168.0.4    imanager.example.com          imanager
192.168.0.5    osp.example.com               osp
192.168.0.6    postgresql.example.com        postgresql
192.168.0.7    identityapps.example.com      identityapps
192.168.0.8    formrenderer.example.com      formrenderer
192.168.0.9    activemq.example.com          activemq
192.168.0.10  identityreporting.example.com  identityreporting
192.168.0.11  sspr.example.com              sspr
```

You must also add the following entries on the hosts file of the machine where you will access the containers from:

```
<IP Address of Docker host A> <FQDN of all containers deployed on
Docker Host A> <short name of all containers deployed on Docker
host A>
```

```
<IP Address of Docker host B> <FQDN of all containers deployed on
Docker Host B> <short name of all containers deployed on Docker
host B>
```

NOTE: The examples in the guide assume virtual IP addresses for all the containers. Based on your requirement, you can assign IP addresses that are accessible across your network.

- ◆ You must know the ports that you want to use for each containers in your deployment. You must expose the required ports and map the container ports with the ports on the Docker host. The following table provides information on ports that you must expose on the Docker hosts based on the examples provided in the guide.

Container	Default ports assumed as per the sample deployment
Remote Loader	8090
Fanout Agent	Not applicable
iManager	8743
OSP	8543
Identity Applications	18543
Identity Reporting	28543
Form Renderer	8600

Container	Default ports assumed as per the sample deployment
ActiveMQ	<ul style="list-style-type: none"> ◆ 8161 ◆ 61616
PostgreSQL	5432
SSPR	8443

NOTE: SSPR container runs only on 8443 port.

However, you can customize the ports based on your requirement. The following considerations apply while you expose the ports:

- ◆ Ensure that you expose those ports which are not in use.
- ◆ The container port must be mapped to the same port on the Docker host. For example, the 8543 port on the container must be mapped to the 8543 port on the Docker host.

Creating the Silent Properties File

Identity Manager supports silent mode only for deployment of containers. You must generate the silent properties file if you are deploying containers for the first time. If you are updating containers from previous versions, the silent properties file is not required.

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_idm_conf_generator.tar.gz
```

3 Deploy the container using the following command:

```
docker run --rm -it --name=idm_conf_generator --
hostname=identitymanager.example.com -v /data:/config
idm_conf_generator:idm-4.8.2
```

NOTE

- ◆ Ensure that you specify the machine FQDN as a value for the hostname.
 - ◆ The `--rm` flag deletes the container after the silent properties file is created.
-

4 Specify the silent property file name with the absolute path:

NOTE: Ensure that you create the `silent.properties` file in the `/config` shared volume location. In other words, the silent properties file will be available in the `/data` directory of the Docker host.

5 Specify `n` for the **Do you want to generate inputs for Kubernetes Orchestration** parameter.

6 Decide the Identity Manager server edition you want to install. Enter `y` for Advanced Edition and `n` for Standard Edition.

7 From the list of components available for installation, select the required components:

- ◆ To install Engine, select **Identity Manager Engine**.
- ◆ To install Identity Reporting, select **Identity Reporting**.
- ◆ To install Identity Applications, select **Identity Applications**.

NOTE

- ◆ You must generate a single `silent.properties` file for deploying all the Identity Manager components.
- ◆ Ensure that you specify the following values for the ports used by different containers:

Prompt	Port to be specified
One SSO Server SSL port	8543
Identity Reporting Tomcat HTTPS port	28543
Identity Applications Tomcat HTTPS port	18543

- ◆ Use FQDN for all IP related configuration prompts. In other words, the hostname that you provide in the `/etc/hosts` entry for all components must be specified while generating the `silent.properties` file.
- ◆ The `SSO_SERVER_SSL_PORT`, `TOMCAT_HTTPS_PORT`, `UA_SERVER_SSL_PORT`, and `RPT_TOMCAT_HTTPS_PORT` must be unique ports.

8 (Conditional) If you are deploying containers on a single server using the host network mode, you must perform the following tasks after the `silent.properties` file is generated:

- ◆ Modify the `SSO_SERVER_SSL_PORT` to **8543**, `TOMCAT_HTTPS_PORT` and `UA_SERVER_SSL_PORT` to **18543**, and `RPT_TOMCAT_HTTPS_PORT` to **28543** respectively.
- ◆ Add the `SKIP_PORT_CHECK=1` entry.

NOTE: When the `silent.properties` file is generated, it will be available in the shared volume of your Docker host. For example, `/data`.

Deploying Containers on a Single Server

In this example, all the Identity Manager containers are deployed on a single Docker host using the host network mode.

The containers must be deployed in the following order:

- ◆ [“Deploying Identity Manager Engine Container” on page 18](#)
- ◆ [“Deploying Remote Loader Container” on page 18](#)
- ◆ [“Deploying Fanout Agent Container” on page 19](#)
- ◆ [“Deploying iManager Container” on page 19](#)
- ◆ [“Generating Certificate With Identity Vault Certificate Authority” on page 21](#)
- ◆ [“Deploying OSP Container” on page 22](#)

- ♦ “Deploying PostgreSQL Container” on page 23
- ♦ “Deploying Identity Applications Container” on page 24
- ♦ “Deploying Form Renderer Container” on page 25
- ♦ “Deploying ActiveMQ Container” on page 26
- ♦ “Deploying Identity Reporting Container” on page 26
- ♦ “Deploying SSPR Container” on page 27

Deploying Identity Manager Engine Container

1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

3 Run the following command to load the image:

```
docker load --input IDM_482_identityengine.tar.gz
```

4 Deploy the container using the following command:

```
docker run -d --network=host --name=engine-container -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 identityengine:idm-4.8.2
```

5 To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/idm/log/idmconfigure.log
```

6 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it engine-container bash
```

NOTE: To run the Identity Vault utilities such as `ndstrace` or `ndsrepair`, log in to the container as a non-root user called as `nds`. These utilities cannot be run if you are logged in as a root user. To log in to the container as a `nds` user, run the `docker exec -it engine-container su nds` command.

Deploying Remote Loader Container

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_remoteloader.tar.gz
```

3 Deploy the container using the following command:

```
docker run -d --network=host --name=rl-container -v /data:/config --
stop-timeout 100 remoteloader:idm-4.8.2
```

The driver files can be found at the `/opt/novell/eDirectory/lib/dirxml/classes/` directory of the container.

NOTE: The 32-bit Remote Loader is not supported with containers.

- 4 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it rl-container bash
```

- 5 Configure Remote Loader. For more information, see [Configuring the Remote Loader and Drivers](#) in the *NetIQ Identity Manager Driver Administration Guide*.
- 6 Ensure that the configuration file is available in the `/config` shared volume of the container. For example, `config8000.txt`.

Deploying Fanout Agent Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_fanoutagent.tar.gz
```

- 3 Deploy the container using the following command:

```
docker run -d --network=host --name=foa-container -v /data:/config --
stop-timeout 100 fanoutagent:idm-4.8.2
```

- 4 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it foa-container bash
```

- 5 Configure the Fanout Agent. For more information, see [Configuring the Fanout Agent](#) in the *NetIQ Identity Manager Driver for JDBC Fanout Implementation Guide*.

Deploying iManager Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_iManager323.tar.gz
```

- 3 Create a `.env` file with the required configuration to suit your environment. For example, the `iManager.env` is created in the `/data` directory.

```

# Certificate Public Key Algorithm
# Allowed Values: RSA, ECDSA256, ECDSA384
CERTIFICATE_ALGORITHM=RSA
# Cipher Suite
# Allowed Values:
# For RSA - NONE, LOW, MEDIUM HIGH
# For ECDSA256 - SUITEB128ONLY
# For ECDSA384 - SUITEB128, SUITEB192
CIPHER_SUITE=NONE
# Tomcat Server HTTP Port
TOMCAT_HTTP_PORT=8080
# Tomcat Server SSL Port
TOMCAT_SSL_PORT=8743
# iManager Authorized User (admin_name.container_name.tree_name)
AUTHORIZED_USER=

```

4 Create a sub-directory called as iManager under the shared volume /data.

5 Deploy the container using the following command:

```

docker run -d --network=host --name=iman-container -v /data:/config -v
/data/iManager.env:/etc/opt/novell/iManager/conf/iManager.env --stop-
timeout 100 imanager:3.2.3

```

6 To install the Identity Manager plug-ins, perform the following steps:

6a Log in to iManager.

```
https://identitymanager.example.com:8743/nps/
```

6b Click **Configure**.

6c Click **Plug-in Installation** and then click **Available NetIQ Plug-in Modules**.

6d Select all the plug-ins from the **NetIQ Plug-in Modules** list and then click **Install**.

To obtain the plug-ins offline, perform the following steps:

1. Download the `Identity_Manager_4.8.2_Linux.iso` from the NetIQ Downloads website.
2. Mount the downloaded `.iso`.
3. From the mounted location, navigate to the `/iManager/plugins` directory and obtain the required plug-ins.

Alternatively, you can install the plug-ins from the [iManager plug-ins website](#).

7 Restart the iManager container.

```
docker restart iman-container
```

8 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

For more information about deploying the iManager container, see the [Deploying iManager Using Docker Container](#) in the *NetIQ iManager Installation Guide*.

Generating Certificate With Identity Vault Certificate Authority

(Conditional) This section applies only if you are using Identity Vault as the Certificate Authority.

The following components require you to generate certificate before they are deployed. Before you generate the certificates for the following components, ensure that you deploy the [Identity Manager Engine](#) and [iManager](#) containers.

- ♦ OSP
- ♦ Identity Applications
- ♦ Identity Reporting

Perform the following steps to generate the certificate:

- 1 Log in to the iManager container.

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

- 2 Ensure that you set the Java path. For example, run the following command:

```
export PATH=<java installed location>/bin:$PATH
```

For example,

```
export PATH=/opt/netiq/common/jre/bin/:$PATH
```

NOTE: Ensure that the Java version installed is Azul Zulu 1.80_265 or later.

- 3 Generate the PKCS keystore:

```
keytool -genkey -alias idm -keyalg RSA -storetype pkcs12 -keystore /  
config/tomcat.ks -validity 3650 -keysize 2048 -dname  
"CN=identitymanager.example.com" -keypass <password> -storepass  
<password>
```

- 4 Generate a certificate signing request:

```
keytool -certreq -v -alias idm -file /config/idm.csr -keypass  
<password> -keystore /config/tomcat.ks -storepass <password>
```

- 5 Generate a self-signed certificate:

- 5a Launch iManager from Docker host and log in as an administrator.

- 5b Navigate to **Roles and Tasks > NetIQ Certificate Server > Issue Certificate**.

- 5c Browse to the `.csr` file created in step 3. For example, `idm.csr`.

- 5d Click **Next**.

- 5e Specify the key usage and click **Next**.

- 5f For the certificate type, select **Unspecified**.

- 5g Click **Next**.

- 5h Specify the validity of the certificate and click **Next**.

- 5i Select the **File in binary DER format radio** button.

- 5j Click **Next**.

- 5k** Click **Finish**.
- 5l** Download the certificate and copy the downloaded certificate to the /data directory.
- 6** Export the root certificate in .der format:
- 6a** Launch iManager from Docker host and log in as an administrator.
 - 6b** Navigate to **Roles and Tasks > NetIQ Certificate Access > Server Certificates**.
 - 6c** Select the **SSL CertificateDNS** check box and click **Export**.
 - 6d** In the **Certificates** drop-down list, select the Organizational CA.
 - 6e** In the **Export Format** drop-down list, select DER.
 - 6f** Click **Next**.
 - 6g** Download the certificate and copy the downloaded certificate to the /data directory.
- 7** Import the certificates into the PKCS keystore you created in step 2:
- ```
keytool -import -trustcacerts -alias root -keystore /config/tomcat.ks -
file /config/cert.der -storepass <password> -noprompt

keytool -import -alias idm -keystore /config/tomcat.ks -file /config/
idm.der -storepass <password> -noprompt
```

---

**NOTE:** Ensure that the keystore is available in the path that was specified as an input for deployment.

---

## Deploying OSP Container

---

**NOTE:** Before you deploy the OSP container, ensure that you generate the required certificate. For more information, see [Generating Certificate With Identity Vault Certificate Authority](#).

---

- 1** Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.
- 2** Ensure that the `SSO_SERVER_SSL_PORT` property is set to a unique port.
- 3** From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 4** Run the following command to load the image:

```
docker load --input IDM_482_osp.tar.gz
```
- 5** Deploy the container using the following command:

```
docker run -d --network=host --name=osp-container -v /data:/config -e
SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100
osp:idm-4.8.2
```
- 6** To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/osp/log/idmconfigure.log
```
- 7** Stop the container using the following command:

```
docker stop osp-container
```

- 8 Run the following command to modify the Tomcat shutdown port in the `server.xml` file. In the following example, the port 8005 will be changed to 18005:

```
sed -i "s~8005~18005~g" /data/osp/tomcat/conf/server.xml
```

- 9 Start the container using the following command:

```
docker start osp-container
```

- 10 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it osp-container bash
```

- 11 Navigate to the `/opt/netiq/idm/apps/configupdate/` directory.

- 12 Modify the `configupdate.sh.properties` file.

- 13 Set the value of the `no_nam_oauth` parameter to *false*.

- 14 Save the `configupdate.sh.properties` file.

- 15 Run the following command to exit the container.

```
exit
```

## Deploying PostgreSQL Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_postgres.tar.gz
```

- 3 Create a sub-directory under the shared volume `/data`, for example, `postgres`.

```
mkdir postgres
```

- 4 Deploy the container using the following command:

```
docker run -d --network=host --name=postgresql-container -e
POSTGRES_PASSWORD=<password> -v /data/postgres:/var/lib/postgresql/data
--stop-timeout 100 postgres:12.2-alpine
```

For example,

```
docker run -d --network=host --name=postgresql-container -e
POSTGRES_PASSWORD=novell -v /data/postgres:/var/lib/postgresql/data --
stop-timeout 100 postgres:12.2-alpine
```

- 5 Create the `idmdamin` user for Identity Applications.

```
docker exec -it postgresql-container psql -U postgres -c "CREATE USER
idmadmin WITH ENCRYPTED PASSWORD '<password>'"
```

- 6 Create the Identity Applications, Workflow, and Identity Reporting databases.

```
docker exec -it postgresql-container psql -U postgres -c "CREATE
DATABASE idmuserappdb"
```

```
docker exec -it postgresql-container psql -U postgres -c "CREATE
DATABASE igaworkflowdb"
```

```
docker exec -it postgresql-container psql -U postgres -c "CREATE
DATABASE idmrptdb"
```

---

**NOTE:** These databases are used while you configure the Identity Applications and Identity Reporting containers.

---

- 7 Grant all the privileges on the databases for the idmadmin user:

```
docker exec -it postgresql-container psql -U postgres -c "GRANT ALL
PRIVILEGES ON DATABASE idmuserappdb TO idmadmin"
```

```
docker exec -it postgresql-container psql -U postgres -c "GRANT ALL
PRIVILEGES ON DATABASE igaworkflowdb TO idmadmin"
```

- 8 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it postgresql-container bash
```

## Deploying Identity Applications Container

---

**NOTE:** Before you deploy the Identity Applications container, ensure that you generate the required certificate. For more information, see [Generating Certificate With Identity Vault Certificate Authority](#).

---

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.
- 2 Ensure that the `UA_SERVER_SSL_PORT` property is set to a unique port.
- 3 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 4 Run the following command to load the image:

```
docker load --input IDM_482_identityapplication.tar.gz
```

- 5 Deploy the container using the following command:

```
docker run -d --network=host --name=idapps-container -v /data:/config -
e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100
identityapplication:idm-4.8.2
```

- 6 To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/userapp/log/idmconfigure.log
```

- 7 Run the following command to log in to the container.

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it idapps-container bash
```

- 8 Run the following command:



---

**NOTE:** Before performing this step, ensure that the container is deployed successfully.

---

```
/opt/netiq/common/jre/bin/keytool -importkeystore -srckeystore /config/tomcat.ks -srcstorepass <password> -destkeystore /opt/netiq/idm/apps/tomcat/conf/idm.jks -deststorepass <password>
```

- 9 Run the following command to exit the container.

```
exit
```

- 10 Run the following command to modify the Tomcat shutdown port in the `server.xml` file. In the following example, the port 8005 will be changed to 28005:

```
sed -i "s~8005~28005~g" /data/userapp/tomcat/conf/server.xml
```

- 11 Restart the container using the following command:

```
docker restart idapps-container
```

---

**NOTE:** To modify any settings in the configuration update utility, launch `configupdate.sh` from the `/opt/netiq/idm/apps/configupdate/` directory of the Identity Applications container. The configuration update utility can be launched in console mode only.

---

## Deploying Form Renderer Container

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_formrenderer.tar.gz
```

- 4 Deploy the container using the following command:

```
docker run -d --network=host --name=fr-container -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 formrenderer:itm-4.8.2
```

- 5 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it fr-container bash
```

## Deploying ActiveMQ Container

---

**NOTE:** This procedure assumes that you will use the ActiveMQ container with the Identity Applications container. To use the ActiveMQ container with the Fanout Agent container, you must deploy a new instance of the ActiveMQ container with different IP address and ports.

---

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:  

```
docker load --input IDM_482_activemq.tar.gz
```
- 3 Deploy the container using the following command:  

```
docker run -d --network=host --name=amq-container -v /data:/config --env-file /data/silent.properties --stop-timeout 100 activemq:itm-4.8.2
```
- 4 To log in to the container, run the following command:  

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it amq-container bash
```

## Deploying Identity Reporting Container

---

**NOTE:** Before you deploy the Identity Reporting container, ensure that you generate the required certificate. For more information, see [Generating Certificate With Identity Vault Certificate Authority](#).

---

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.
- 2 Ensure that the `TOMCAT_HTTPS_PORT` property is set to a unique port.
- 3 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 4 Run the following command to load the image:  

```
docker load --input IDM_482_identityreporting.tar.gz
```
- 5 Deploy the container using the following command:  

```
docker run -d --network=host --name=rpt-container -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 identityreporting:itm-4.8.2
```
- 6 To verify whether the container was successfully deployed, check the log files by running the following command:  

```
tail -f /data/reporting/log/idmconfigure.log
```
- 7 Run the following command to log in to the container:  

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it rpt-container bash
```

- 8 Run the following command:

---

**NOTE:** Before performing this step, ensure that the container is deployed successfully.

---

```
/opt/netiq/common/jre/bin/keytool -importkeystore -srckeystore /config/tomcat.ks -srcstorepass <password> -destkeystore /opt/netiq/idm/apps/tomcat/conf/idm.jks -deststorepass <password>
```

- 9 Run the following command to exit the container.

```
exit
```

- 10 Run the following command to modify the Tomcat shutdown port in the `server.xml` file. In the following example, the port 8005 will be changed to 38005:

```
sed -i "s~8005~38005~g" /data/reporting/tomcat/conf/server.xml
```

- 11 (Conditional) Applies only if you are using Identity Vault as the Certificate Authority.

Add the `-Dcom.sun.net.ssl.checkRevocation=false` parameter in the `export CATALINA_OPTS` entry of the `setenv.sh` file. In this example, the `setenv.sh` file is located under the `/data/reporting/tomcat/bin/` directory.

- 12 Restart the container using the following command:

```
docker restart rpt-container
```

## Deploying SSPR Container

Perform the following tasks to deploy the SSPR container:

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

- 2 Create a sub-directory under the shared volume `/data`, for example, `sspr`.

```
mkdir sspr
```

- 3 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 4 Run the following command to load the image:

```
docker load --input IDM_482_sspr.tar.gz
```

- 5 Deploy the container using the following command:

```
docker run -d --network=host --name=sspr-container -v /data/sspr:/config --stop-timeout 100 sspr/sspr-webapp:latest
```

- 6 Run the following command from the Docker host to copy the `silent.properties` file from the Docker host to SSPR container:

```
docker cp /data/silent.properties sspr-container:/tmp
```

- 7 Load the silent properties file to the SSPR container.


```
docker exec -it sspr-container /app/command.sh ImportPropertyConfig /tmp/silent.properties
```

---

**NOTE:** Check if the `SSPRConfiguration.xml` is created under the `/config` directory of SSPR container and verify the content of the file.

---

**8** Import the OAuth certificate to SSPR:

- 8a** From the Docker host, edit the `SSPRConfiguration.xml` file located at `/data/sspr/` directory and set the value of the `configIsEditable` flag to **true** and save the changes.
- 8b** Launch a browser and enter the `https://identitymanager.example.com:8443/sspr` URL.
- 8c** Click **OK**.
- 8d** Log in using administrator credentials, for example, `uaadmin`.
- 8e** Click on the user, for example, `uaadmin`, on the top-right corner and then click **Configuration Editor**.
- 8f** Specify the configuration password and click **Sign In**.
- 8g** Click **Settings > Single Sign On (SSO) Client > OAuth** and ensure that all URLs use the HTTPS protocol and correct ports.
- 8h** Under **OAuth Server Certificate**, click **Import from Server** to import a new certificate and then click **OK**.
- 8i** Click  at the top-right corner to save the certificate.
- 8j** Review the changes and click **OK**.
- 8k** After the SSPR application is restarted, edit the `SSPRConfiguration.xml` file and set the value of the `configIsEditable` flag to **false** and save the changes.

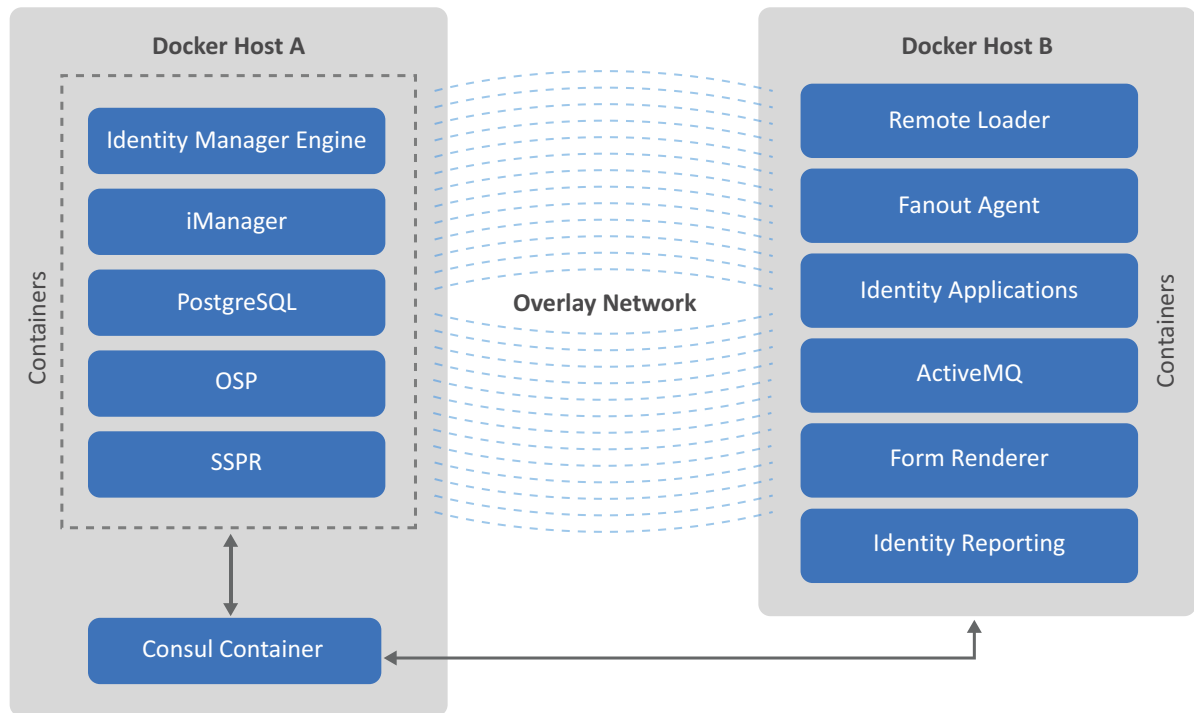
## Deploying Containers on Distributed Servers

NetIQ recommends you to use overlay or bridge network mode for deploying all Identity Manager containers in a distributed setup. The scenarios documented in the guide provide instructions and commands to deploy containers in a overlay network. However, you can also use bridge network for deploying containers.

In the following distributed servers scenario, we will deploy the Identity Manager Engine, iManager, PostgreSQL, OSP, and SSPR containers on Docker Host A. On Docker Host B, we will deploy the Remote Loader, Fanout Agent, Identity Applications, ActiveMQ, Form Renderer, and Identity Reporting containers. We will deploy the Consul container on Docker host A. However, you can deploy the Consul container on any of the Docker hosts in your deployment.

The following figure illustrates the deployment of Identity Manager containers on two Docker hosts in a overlay network.

**Figure 2-1** Containers Deployment Architecture in an Overlay Network



Perform the following steps to set up an overlay network:

- 1 Run the following command on Docker Host A:

```
docker run -d -p <host port>:8500 -h consul --name <container name> --restart unless-stopped progrium/consul -server -bootstrap
```

For example:

```
docker run -d -p 8500:8500 -h consul --name consul --restart unless-stopped progrium/consul -server -bootstrap
```

- 2 On both the Docker Hosts, edit the **docker** file located at `/etc/sysconfig/` directory and add the following line:

```
DOCKER_OPTS="-H tcp://0.0.0.0:2375 -H unix:///var/run/docker.sock --cluster-advertise <Master Server Network Interface>:2375 --cluster-store consul://<Docker Host A IP Address>:<Docker Host A Port>"
```

For example:

```
DOCKER_OPTS="-H tcp://0.0.0.0:2375 -H unix:///var/run/docker.sock --cluster-advertise eth0:2375 --cluster-store consul://172.120.0.1:8500"
```

- 3 Restart the Docker service on both the Docker hosts:

```
systemctl restart docker
```

- 4 On Docker Host B, run the following command to check whether Docker Host B is added to the cluster:

```
docker info
```

The sample output will be as follows:

```
Cluster store: consul://<Docker HOST A IP Address>:8500
```

```
Cluster advertise: <Docker HOST B IP Address>:2375
```

**5 Create an overlay network on any of the Docker hosts:**

```
docker network create -d overlay --subnet=<subnet in CID format that represents a network segment> --gateway=<ipv4 gateway> <name of the overlay network>
```

For example:

```
docker network create -d overlay --subnet=192.168.0.0/24 --gateway=192.168.0.1 idmoverlaynetwork
```

**6 Run the following command to verify whether the overlay network is created:**

```
docker network ls
```

The containers must be deployed in the following order:

- ♦ “Deploying Identity Manager Engine Container” on page 30
- ♦ “Deploying Remote Loader Container” on page 31
- ♦ “Deploying Fanout Agent Container” on page 31
- ♦ “Deploying iManager Container” on page 32
- ♦ “Generating Certificates With Identity Vault Certificate Authority” on page 33
- ♦ “Deploying OSP Container” on page 37
- ♦ “Deploying PostgreSQL Container” on page 38
- ♦ “Deploying Identity Applications Container” on page 39
- ♦ “Deploying Form Renderer Container” on page 40
- ♦ “Deploying ActiveMQ Container” on page 41
- ♦ “Deploying Identity Reporting Container” on page 41
- ♦ “Deploying SSPR Container” on page 42

## Deploying Identity Manager Engine Container

**1** Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

**2** From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

**3** Run the following command to load the image:

```
docker load --input IDM_482_identityengine.tar.gz
```

**4** Deploy the container using the following command:

```
docker run -d --ip=192.168.0.12 --network=idmoverlaynetwork --hostname=identityengine.example.com --name=engine-container -v /etc/hosts:/etc/hosts -v /data:/config -p 8028:8028 -p 524:524 -p 389:389 -p 8030:8030 -p 636:636 -e SILENT_INSTALL_FILE=/config/silent.properties -stop-timeout 100 identityengine:idm-4.8.2
```

**5** To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/idm/log/idmconfigure.log
```

- 6 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it engine-container bash
```

---

**NOTE:** To run the Identity Vault utilities such as `ndstrace` or `ndsrepair`, log in to the container as a non-root user called as `nds`. These utilities cannot be run if you are logged in as a root user. To log in to the container as a `nds` user, run the `docker exec -it engine-container su nds` command.

---

## Deploying Remote Loader Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_remoteloader.tar.gz
```

- 3 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.2 --network=idmoverlaynetwork --hostname=remoteloader.example.com -p 8090:8090 --name=rl-container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 remoteloader:idm-4.8.2
```

The driver files can be found at the `/opt/novell/eDirectory/lib/dirxml/classes/` directory of the container.

- 4 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it rl-container bash
```

- 5 Configure Remote Loader. For more information, see [Configuring the Remote Loader and Drivers](#) in the *NetIQ Identity Manager Driver Administration Guide*.

- 6 Ensure that the configuration file is available in the `/config` shared volume of the container. For example, `config8000.txt`.

## Deploying Fanout Agent Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_fanoutagent.tar.gz
```

- 3 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.3 --network=idmoverlaynetwork --
hostname=fanoutagent.example.com --name=foa-container -v /etc/hosts:/
etc/hosts -v /data:/config --stop-timeout 100 fanoutagent:idm-4.8.2
```

- 4 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it foa-container bash
```

- 5 Configure the Fanout Agent. For more information, see [Configuring the Fanout Agent](#) in the *NetIQ Identity Manager Driver for JDBC Fanout Implementation Guide*.

## Deploying iManager Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_iManager323.tar.gz
```

- 3 Create a `.env` file with the required configuration to suit your environment. For example, the `iManager.env` is created in the `/data` directory.

```
Certificate Public Key Algorithm
Allowed Values: RSA, ECDSA256, ECDSA384
CERTIFICATE_ALGORITHM=RSA
Cipher Suite
Allowed Values:
For RSA - NONE, LOW, MEDIUM HIGH
For ECDSA256 - SUITEB128ONLY
For ECDSA384 - SUITEB128, SUITEB192
CIPHER_SUITE=NONE
Tomcat Server HTTP Port
TOMCAT_HTTP_PORT=8080
Tomcat Server SSL Port
TOMCAT_SSL_PORT=8743
iManager Authorized User (admin_name.container_name.tree_name)
AUTHORIZED_USER=
```

- 4 Create a sub-directory called as `iManager` under the shared volume `/data`.

- 5 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.4 --name=iman-container --
network=idmoverlaynetwork --hostname=imanager.example.com -v /etc/
hosts:/etc/hosts -v /data:/config -v /data/iManager.env:/etc/opt/
novell/iManager/conf/iManager.env -p 8743:8743 --stop-timeout 100
imanager:3.2.3
```

- 6 To install the Identity Manager plug-ins, perform the following steps:

- 6a Log in to iManager.

```
https://imanager.example.com:8743/nps/
```

- 6b Click **Configure**.



**6c** Click **Plug-in Installation** and then click **Available NetIQ Plug-in Modules**.

**6d** Select all the plug-ins from the **NetIQ Plug-in Modules** list and then click **Install**.

To obtain the plug-ins offline, perform the following steps:

1. Download the `Identity_Manager_4.8.2_Linux.iso` from the NetIQ Downloads website.
2. Mount the downloaded `.iso`.
3. From the mounted location, navigate to the `/iManager/plugins` directory and obtain the required plug-ins.

Alternatively, you can install the plug-ins from the [iManager plug-ins website](#).

**7** Restart the iManager container.

```
docker restart iman-container
```

**8** To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

For more information about deploying the iManager container, see the [Deploying iManager Using Docker Container](#) in the *NetIQ iManager Installation Guide*.

## Generating Certificates With Identity Vault Certificate Authority

*(Conditional) This section applies only if you are using Identity Vault as the Certificate Authority.*

The following components require you to generate certificates before they are deployed. Before you generate the certificates for the following components, ensure that you deploy the [Identity Manager Engine](#) and [iManager](#) containers.

- ♦ [OSP](#)
- ♦ [Identity Applications](#)
- ♦ [Identity Reporting](#)

## Generating Certificates for OSP

Perform the following steps to generate the certificates:

**1** Log in to the iManager container.

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

**2** Ensure that you set the Java path. For example, run the following command:

```
export PATH=<java installed location>/bin:$PATH
```

For example,

```
export PATH=/opt/netiq/common/jre/bin/:$PATH
```

---

**NOTE:** Ensure that the Java version installed is Azul Zulu 1.80\_265 or later.

---

**3** Generate the PKCS keystore:

```
keytool -genkey -alias osp -keyalg RSA -storetype pkcs12 -keystore /
config/tomcat-osp.ks -validity 3650 -keysize 2048 -dname
"CN=osp.example.com" -keypass <password> -storepass <password>
```

**4** Generate a certificate signing request:

```
keytool -certreq -v -alias osp -file /config/osp.csr -keypass
<password> -keystore /config/tomcat-osp.ks -storepass <password>
```

**5** Generate a self-signed certificate:

**5a** Launch iManager from Docker host and log in as an administrator.

**5b** Navigate to **Roles and Tasks > NetIQ Certificate Server > Issue Certificate**.

**5c** Browse to the `.csr` file created in step 3. For example, `osp.csr`.

**5d** Click **Next**.

**5e** Specify the key usage and click **Next**.

**5f** For the certificate type, select **Unspecified**.

**5g** Click **Next**.

**5h** Specify the validity of the certificate and click **Next**.

**5i** Select the **File in binary DER format radio** button.

**5j** Click **Next**.

**5k** Click **Finish**.

**5l** Download the certificate and copy the downloaded certificate to the `/data` directory.

**6** Export the root certificate in `.der` format:

**6a** Launch iManager from Docker host and log in as an administrator.

**6b** Navigate to **Roles and Tasks > NetIQ Certificate Access > Server Certificates**.

**6c** Select the **SSL CertificateDNS** check box and click **Export**.

**6d** In the **Certificates** drop-down list, select the Organizational CA.

**6e** In the **Export Format** drop-down list, select **DER**.

**6f** Click **Next**.

**6g** Download the certificate and copy the downloaded certificate to the `/data` directory.

**7** Import the certificates into the PKCS keystore you created in step 2:

```
keytool -import -trustcacerts -alias root -keystore /config/tomcat-
osp.ks -file /config/cert.der -storepass <password> -noprompt

keytool -import -alias osp -keystore /config/tomcat-osp.ks -file /
config/osp.der -storepass <password> -noprompt
```

---

**NOTE:** Ensure that the keystore is available in the path that was specified as an input for deployment.

---

## Generating Certificates for Identity Applications

Perform the following steps to generate the certificates:

- 1 Log in to the iManager container.

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

- 2 Ensure that you set the Java path. For example, run the following command:

```
export PATH=<java installed location>/bin:$PATH
```

For example,

```
export PATH=/opt/netiq/common/jre/bin/:$PATH
```

---

**NOTE:** Ensure that the Java version installed is Azul Zulu 1.80\_265 or later.

---

- 3 Generate the PKCS keystore:

```
keytool -genkey -alias ua -keyalg RSA -storetype pkcs12 -keystore /
config/tomcat-ua.ks -validity 3650 -keysize 2048 -dname
"CN=identityapps.example.com" -keypass <password> -storepass <password>
```

- 4 Generate a certificate signing request:

```
keytool -certreq -v -alias ua -file /config/ua.csr -keypass <password>
-keystore /config/tomcat-ua.ks -storepass <password>
```

- 5 Generate a self-signed certificate:

- 5a Log in to iManager as an administrator.

- 5b Navigate to **Roles and Tasks > NetIQ Certificate Server > Issue Certificate**.

- 5c Browse to the `.csr` file created in step 3. For example, `ua.csr`.

- 5d Click **Next**.

- 5e Specify the key usage and click **Next**.

- 5f For the certificate type, select **Unspecified**.

- 5g Click **Next**.

- 5h Specify the validity of the certificate and click **Next**.

- 5i Select the **File in binary DER format radio** button.

- 5j Click **Next**.

- 5k Click **Finish**.

- 5l Download the certificate and copy the downloaded certificate to the `/data` directory.

- 6 Export the root certificate in `.der` format:

- 6a Log in to iManager as an administrator.

- 6b Navigate to **Roles and Tasks > NetIQ Certificate Access > Server Certificates**.

- 6c Select the **SSL CertificateDNS** check box and click **Export**.

- 6d In the **Certificates** drop-down list, select the Organizational CA.

- 6e In the Export Format drop-down list, select DER.

**6f** Click **Next**.

**6g** Download the certificate and copy the downloaded certificate to the /data directory.

**7** Import the certificates into the PKCS keystore in step 2:

```
keytool -import -trustcacerts -alias root -keystore /config/tomcat-ua.ks -file /config/cert.der -storepass <password> -noprompt
```

```
keytool -import -alias ua -keystore /config/tomcat-ua.ks -file /config/ua.der -storepass <password> -noprompt
```

---

**NOTE:** Ensure that the certificates are available in the path that was specified as an input for deployment.

---

## Generating Certificates for Identity Reporting

Perform the following steps to generate the certificates:

**1** Log in to the iManager container.

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it iman-container bash
```

**2** Ensure that you set the Java path. For example, run the following command:

```
export PATH=<java installed location>/bin:$PATH
```

For example,

```
export PATH=/opt/netiq/common/jre/bin/:$PATH
```

---

**NOTE:** Ensure that the Java version installed is Azul Zulu 1.80\_265 or later.

---

**3** Generate the PKCS keystore:

```
keytool -genkey -alias rpt -keyalg RSA -storetype pkcs12 -keystore /config/tomcat-rpt.ks -validity 3650 -keysize 2048 -dname "CN=identityreporting.example.com" -keypass <password> -storepass <password>
```

**4** Generate a certificate signing request:

```
keytool -certreq -v -alias rpt -file /config/rpt.csr -keypass <password> -keystore /config/tomcat-rpt.ks -storepass <password>
```

**5** Generate a self-signed certificate:

**5a** Log in to iManager as an administrator.

**5b** Navigate to **Roles and Tasks > NetIQ Certificate Server > Issue Certificate**.

**5c** Browse to the .csr file created in step 3. For example, rpt.csr.

**5d** Click **Next**.

**5e** Specify the key usage and click **Next**.

**5f** For the certificate type, select **Unspecified**.

**5g** Click **Next**.

- 5h Specify the validity of the certificate and click **Next**.
  - 5i Select the **File in binary DER format radio** button.
  - 5j Click **Next**.
  - 5k Click **Finish**.
  - 5l Download the certificate and copy the downloaded certificate to the /data directory.
- 6 Export the root certificate in .der format:
- 6a Log in to iManager as an administrator.
  - 6b Navigate to **Roles and Tasks > NetIQ Certificate Access > Server Certificates**.
  - 6c Select the **SSL CertificateDNS** check box and click **Export**.
  - 6d In the **Certificates** drop-down list, select the Organizational CA.
  - 6e In the Export Format drop-down list, select DER.
  - 6f Click **Next**.
  - 6g Download the certificate and copy the downloaded certificate to the /data directory.
- 7 Import the certificates into the PKCS keystore you created in step 2:
- ```
keytool -import -trustcacerts -alias root -keystore /config/tomcat-rpt.ks -file /config/cert.der -storepass <password> -noprompt
keytool -import -alias rpt -keystore /config/tomcat-rpt.ks -file /config/rpt.der -storepass <password> -noprompt
```

NOTE: Ensure that the certificates are available in the path that was specified as an input for deployment.

Deploying OSP Container

NOTE: Before you deploy the OSP container, ensure that you generate the required certificates. For more information, see [Generating Certificates for OSP](#).

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.
- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 3 Run the following command to load the image:


```
docker load --input IDM_482_osp.tar.gz
```
- 4 Deploy the container using the following command:


```
docker run -d --ip=192.168.0.5 --network=idmoverlaynetwork --hostname=osp.example.com -p 8543:8543 --name=osp-container -v /etc/hosts:/etc/hosts -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 osp:idm-4.8.2
```
- 5 To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/osp/log/idmconfigure.log
```

- 6 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it osp-container bash
```

- 7 Navigate to the `/opt/netiq/idm/apps/configupdate/` directory.

- 8 Modify the `configupdate.sh.properties` file.

- 9 Set the value of the `no_nam_oauth` parameter to `false`.

- 10 Save the `configupdate.sh.properties` file.

- 11 Run the following command to exit the container.

```
exit
```

Deploying PostgreSQL Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_postgres.tar.gz
```

- 3 Create a sub-directory under the shared volume `/data`, for example, `postgres`.

```
mkdir postgres
```

- 4 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.6 --network=idmoverlaynetwork --hostname=postgresql.example.com --name=postgresql-container -p 5432:5432 -e POSTGRES_PASSWORD=<password> -v /data/postgres:/var/lib/postgresql/data -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 postgres:12.2-alpine
```

For example,

```
docker run -d --ip=192.168.0.6 --network=idmoverlaynetwork --hostname=postgresql.example.com --name=postgresql-container -p 5432:5432 -e POSTGRES_PASSWORD=novell -v /data/postgres:/var/lib/postgresql/data -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 postgres:12.2-alpine
```

- 5 Create the `idmdamin` user for Identity Applications.

```
docker exec -it postgresql-container psql -U postgres -c "CREATE USER idmadmin WITH ENCRYPTED PASSWORD '<password>'"
```

- 6 Create the Identity Applications, Workflow, and Identity Reporting databases.

```
docker exec -it postgresql-container psql -U postgres -c "CREATE DATABASE idmuserappdb"
```

```
docker exec -it postgresql-container psql -U postgres -c "CREATE DATABASE igaworkflowdb"
```

```
docker exec -it postgresql-container psql -U postgres -c "CREATE
DATABASE idmrptdb"
```

NOTE: These databases are used while you configure the Identity Applications and Identity Reporting containers.

- 7 Grant all the privileges on the databases for the idmadmin user:

```
docker exec -it postgresql-container psql -U postgres -c "GRANT ALL
PRIVILEGES ON DATABASE idmuserappdb TO idmadmin"
```

```
docker exec -it postgresql-container psql -U postgres -c "GRANT ALL
PRIVILEGES ON DATABASE igaworkflowdb TO idmadmin"
```

- 8 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it postgresql-container bash
```

Deploying Identity Applications Container

NOTE: Before you deploy the Identity Applications container, ensure that you generate the required certificates. For more information, see [Generating Certificates for Identity Applications](#).

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

NOTE: Specify the exposed port, 18543, as the value for the application server port.

- 2 From the location where you have extracted the Identity_Manager_4.8.2_Containers.tar.gz file, navigate to the Identity_Manager_4.8.2_Containers directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_identityapplication.tar.gz
```

- 4 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.7 --network=idmoverlaynetwork --
hostname=identityapps.example.com -p 18543:18543 --name=idapps-
container -v /etc/hosts:/etc/hosts -v /data:/config -e
SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100
identityapplication:idm-4.8.2
```

- 5 To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/userapp/log/idmconfigure.log
```

- 6 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it idapps-container bash
```

- 7 Run the following command:

NOTE: Before performing this step, ensure that the container is deployed successfully.

```
/opt/netiq/common/jre/bin/keytool -importkeystore -srckeystore /config/tomcat-osp.ks -srcstorepass <password> -destkeystore /opt/netiq/idm/apps/tomcat/conf/idm.jks -deststorepass <password>
```

- 8 Type `yes` to overwrite the entry for the `root` alias.
- 9 Run the following command to exit the container.

```
exit
```

- 10 Restart the Identity Applications container.

```
docker restart idapps-container
```

NOTE: To modify any settings in the configuration update utility, launch `configupdate.sh` from the `/opt/netiq/idm/apps/configupdate/` directory of the Identity Applications container. The configuration update utility can be launched in console mode only.

Deploying Form Renderer Container

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_formrenderer.tar.gz
```

- 4 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.8 --network=idmoverlaynetwork --hostname=formrenderer.example.com -p 8600:8600 --name=fr-container -v /etc/hosts:/etc/hosts -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 formrenderer:idm-4.8.2
```

- 5 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it fr-container bash
```


Deploying ActiveMQ Container

NOTE: This procedure assumes that you will use the ActiveMQ container with the Identity Applications container. To use the ActiveMQ container with the Fanout Agent container, you must deploy a new instance of the ActiveMQ container with different IP address and ports.

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_activemq.tar.gz
```

- 4 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.9 --network=idmoverlaynetwork --hostname=activemq.example.com -p 8161:8161 -p 61616:61616 --name=amq-container -v /etc/hosts:/etc/hosts -v /data:/config --env-file /data/silent.properties --stop-timeout 100 activemq:idm-4.8.2
```

- 5 To log in to the container, run the following command:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it amq-container bash
```

Deploying Identity Reporting Container

NOTE: Before you deploy the Identity Reporting container, ensure that you generate the required certificates. For more information, see [Generating Certificates for Identity Reporting](#).

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

NOTE: Specify the exposed port, 28543, as the value for the application server port.

- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_identityreporting.tar.gz
```

- 4 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.10 --network=idmoverlaynetwork --hostname=identityreporting.example.com -p 28543:28543 --name=rpt-container -v /etc/hosts:/etc/hosts -v /data:/config -e SILENT_INSTALL_FILE=/config/silent.properties --stop-timeout 100 identityreporting:idm-4.8.2
```

- 5 To verify whether the container was successfully deployed, check the log files by running the following command:

```
tail -f /data/reporting/log/idmconfigure.log
```

- 6 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it rpt-container bash
```

- 7 Run the following command:

NOTE: Before performing this step, ensure that the container is deployed successfully.

```
/opt/netiq/common/jre/bin/keytool -importkeystore -srckeystore /config/tomcat-osp.ks -srcstorepass <password> -destkeystore /opt/netiq/idm/apps/tomcat/conf/idm.jks -deststorepass <password>
```

- 8 Type *yes* to overwrite the entry for the `root` alias.

- 9 Run the following command to exit the container.

```
exit
```

- 10 Restart the Identity Reporting container.

```
docker restart rpt-container
```

Deploying SSPR Container

Perform the following tasks to deploy the SSPR container:

- 1 Use the silent properties file generated in the [Creating the Silent Properties File](#) section for deploying the container.

- 2 Create a sub-directory under the shared volume `/data`, for example, `sspr`.

```
mkdir sspr
```

- 3 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 4 Run the following command to load the image:

```
docker load --input IDM_482_sspr.tar.gz
```

- 5 Deploy the container using the following command:

```
docker run -d --ip=192.168.0.11 --network=idmoverlaynetwork --hostname=sspr.example.com --name=sspr-container -v /etc/hosts:/etc/hosts -v /data/sspr:/config -p 8443:8443 --stop-timeout 100 sspr/sspr-webapp:latest
```

- 6 Run the following command from the Docker host to copy the `silent.properties` file from the Docker host to SSPR container:


```
docker cp /data/silent.properties sspr-container:/tmp
```

- 7 Load the silent properties file to the SSPR container.

```
docker exec -it sspr-container /app/command.sh ImportPropertyConfig /tmp/silent.properties
```

NOTE: Check if the `SSPRConfiguration.xml` is created under the `/config` directory of SSPR container and verify the content of the file.

8 Import the OAuth certificate to SSPR:

- 8a** From the Docker host, edit the `SSPRConfiguration.xml` file located at `/data/sspr` directory and set the value of the `configIsEditable` flag to **true** and save the changes.
- 8b** Launch a browser and enter the `https://sspr.example.com:8443/sspr` URL.
- 8c** Click **OK**.
- 8d** Log in using administrator credentials, for example, `uaadmin`.
- 8e** Click on the user, for example, `uaadmin`, on the top-right corner and then click **Configuration Editor**.
- 8f** Specify the configuration password and click **Sign In**.
- 8g** Click **Settings > Single Sign On (SSO) Client > OAuth** and ensure that all URLs use the HTTPS protocol and correct ports.
- 8h** Under **OAuth Server Certificate**, click **Import from Server** to import a new certificate and then click **OK**.
- 8i** Click  at the top-right corner to save the certificate.
- 8j** Review the changes and click **OK**.
- 8k** After the SSPR application is restarted, edit the `SSPRConfiguration.xml` file and set the value of the `configIsEditable` flag to **false** and save the changes.

3 Updating Identity Manager Containers

This section provides information on updating individual containers of Identity Manager.

The procedures for updating containers are described in subsequent sections.

- ♦ [“Prerequisites for Updating Containers” on page 45](#)
- ♦ [“Updating Containers on a Single Server” on page 45](#)
- ♦ [“Updating Containers on Distributed Servers” on page 51](#)

Prerequisites for Updating Containers

Perform the following steps before you update each of the Identity Manager containers.

IMPORTANT: This section does not apply for the PostgreSQL container. For information about the steps to be performed for the PostgreSQL container, see [Updating PostgreSQL Container](#).

- 1 Take a back up of the shared volume. The examples in the guide assumes `/data` as the shared volume.
- 2 Stop all the Identity Manager containers.

```
docker stop <container name>
```

For example,

```
docker stop engine-container
```
- 3 Delete all the Identity Manager containers.

```
docker rm <container name>
```

For example,

```
docker rm engine-container
```
- 4 (Conditional) Delete all obsolete Docker images.

```
docker rmi <image ID>
```

Updating Containers on a Single Server

The containers must be updated in the following order:

- ♦ [“Updating Identity Manager Engine Container” on page 46](#)
- ♦ [“Updating Remote Loader Container” on page 46](#)
- ♦ [“Updating Fanout Agent Container” on page 47](#)
- ♦ [“Updating iManager Container” on page 47](#)
- ♦ [“Updating OSP Container” on page 48](#)

- ♦ “Updating PostgreSQL Container” on page 48
- ♦ “Updating Identity Applications Container” on page 50
- ♦ “Updating Form Renderer Container” on page 50
- ♦ “Updating ActiveMQ Container” on page 50
- ♦ “Updating Identity Reporting Container” on page 51
- ♦ “Updating SSPR Container” on page 51

Updating Identity Manager Engine Container

- 1 Create a `credentials.properties` file under the shared volume `/data` with the following content.

```
ID_VAULT_ADMIN="<ID_VAULT_ADMIN>"
ID_VAULT_PASSWORD="<ID_VAULT_PASSWORD>"
```

where, `ID_VAULT_ADMIN` must be in dot format.

For example,

```
ID_VAULT_ADMIN="admin.sa.system"
ID_VAULT_PASSWORD="novell"
```

- 2 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 3 Run the following command to load the image:

```
docker load --input IDM_482_identityengine.tar.gz
```

- 4 Update the container using the following command:

```
docker run -d --network=host --name=engine-container -v /etc/hosts:/etc/hosts -v /data:/config -e SILENT_INSTALL_FILE=/config/credentials.properties --stop-timeout 100 identityengine:idm-4.8.2
```

Updating Remote Loader Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_remoteloader.tar.gz
```

- 3 Update the container using the following command:

```
docker run -d --network=host --name=rl-container -v /data:/config --stop-timeout 100 remoteloader:idm-4.8.2
```

The driver files can be found at the `/opt/novell/eDirectory/lib/dirxml/classes/` directory of the container.

- 4 Start the Remote Loader instances.

Updating Fanout Agent Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_fanoutagent.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --network=host --name=foa-container -v /data:/config --stop-timeout 100 fanoutagent:idm-4.8.2
```
- 4 Start Fanout Agent.

Updating iManager Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_iManager323.tar.gz
```
- 3 Ensure that the `iManager.env` file is created and present in the `/data` directory.

```
# Certificate Public Key Algorithm
# Allowed Values: RSA, ECDSA256, ECDSA384
CERTIFICATE_ALGORITHM=RSA
# Cipher Suite
# Allowed Values:
# For RSA - NONE, LOW, MEDIUM HIGH
# For ECDSA256 - SUITEB128ONLY
# For ECDSA384 - SUITEB128, SUITEB192
CIPHER_SUITE=NONE
# Tomcat Server HTTP Port
TOMCAT_HTTP_PORT=8080
# Tomcat Server SSL Port
TOMCAT_SSL_PORT=8743
# iManager Authorized User (admin_name.container_name.tree_name)
AUTHORIZED_USER=
```
- 4 Update the container using the following command:

```
docker run -d --network=host --name=iman-container -v /data:/config -v /data/iManager.env:/etc/opt/novell/iManager/conf/iManager.env --stop-timeout 100 imanager:3.2.3
```
- 5 To install the Identity Manager plug-ins, perform the following steps:
 - 5a Log in to iManager.

```
https://identitymanager.example.com:8743/nps/
```
 - 5b Click **Configure**.
 - 5c Click **Plug-in Installation** and then click **Available NetIQ Plug-in Modules**.
 - 5d Select all the plug-ins from the **NetIQ Plug-in Modules** list and then click **Install**.

To obtain the plug-ins offline, perform the following steps:

1. Download the `Identity_Manager_4.8.2_Linux.iso` from the NetIQ Downloads website.
2. Mount the downloaded `.iso`.
3. From the mounted location, navigate to the `/iManager/plugins` directory and obtain the required plug-ins.

Alternatively, you can install the plug-ins from the [iManager plug-ins website](#).

- 6 Restart the iManager container.

```
docker restart iman-container
```

Updating OSP Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_osp.tar.gz
```

- 3 Update the container using the following command:

```
docker run -d --network=host --name=osp-container -v /data:/config --stop-timeout 100 osp:idm-4.8.2
```

- 4 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it osp-container bash
```

- 5 Navigate to the `/opt/netiq/idm/apps/configupdate/` directory.

- 6 Modify the `configupdate.sh.properties` file.

- 7 Set the value of the `no_nam_oauth` parameter to `false`.

- 8 Save the `configupdate.sh.properties` file.

- 9 Run the following command to exit the container.

```
exit
```

Updating PostgreSQL Container

NOTE

- ♦ *(Conditional) This procedure applies only if you are updating PostgreSQL from the Identity Manager 4.8 version to the 4.8.2 version.*

In other words, do not update the PostgreSQL container if you have already deployed the PostgreSQL version (12.2) shipped with the Identity Manager 4.8.1 containers release. This release does not support any new version of the PostgreSQL container.

- ◆ Before you update the PostgreSQL container, ensure that you stop the dependent containers such as Identity Applications and/or Identity Reporting.
-

- 1 On the Docker host, navigate to any location. For example:

```
cd /tmp
```

- 2 Run the following command to take a back up of the existing PostgreSQL container data.

```
docker exec postgresql-container pg_dumpall -U postgres > dump.sql
```

- 3 Stop the PostgreSQL container.

```
docker stop <container name>
```

For example,

```
docker stop postgresql-container
```

- 4 Delete the PostgreSQL container.

```
docker rm <container name>
```

- 5 Delete the existing PostgreSQL data directory.

```
rm -rf /data/postgres
```

- 6 (Conditional) Delete the PostgreSQL Docker image.

```
docker rmi <image ID>
```

- 7 Create a sub-directory under the shared volume /data, for example, postgres.

```
mkdir postgres
```

- 8 From the location where you have extracted the Identity_Manager_4.8.2_Containers.tar.gz file, navigate to the Identity_Manager_4.8.2_Containers directory.

- 9 Run the following command to load the image:

```
docker load --input IDM_482_postgres.tar.gz
```

- 10 Update the container using the following command:

```
docker run -d --network=host --name=postgresql-container -e  
POSTGRES_PASSWORD=<password> -v /data/postgres:/var/lib/postgresql/data  
--stop-timeout 100 postgres:12.2-alpine
```

For example,

```
docker run -d --network=host --name=postgresql-container -e  
POSTGRES_PASSWORD=novell -v /data/postgres:/var/lib/postgresql/data --  
stop-timeout 100 postgres:12.2-alpine
```

- 11 Copy the data file you backed up on the Docker host ([Step 2](#)) to the new PostgreSQL data directory.

```
cp /tmp/dump.sql /data/postgres
```

- 12 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it postgresql-container bash
```

13 Navigate to the `/var/lib/postgresql/data/` directory.

14 Restore the data backed up in [Step 2](#) to the new PostgreSQL container.

```
psql -U postgres < dump.sql
```

15 Run the following command to exit the container.

```
exit
```

Updating Identity Applications Container

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_identityapplication.tar.gz
```

3 Update the container using the following command:

```
docker run -d --network=host --name=idapps-container -v /data:/config --stop-timeout 100 identityapplication:idm-4.8.2
```

Updating Form Renderer Container

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_formrenderer.tar.gz
```

3 Update the container using the following command:

```
docker run -d --network=host --name=fr-container -v /data:/config --stop-timeout 100 formrenderer:idm-4.8.2
```

Updating ActiveMQ Container

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_activemq.tar.gz
```

3 Update the container using the following command:

```
docker run -d --network=host --name=amq-container -v /data:/config --stop-timeout 100 activemq:idm-4.8.2
```

Updating Identity Reporting Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_identityreporting.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --network=host --name=rpt-container -v /data:/config --stop-timeout 100 identityreporting:idm-4.8.2
```

Updating SSPR Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_sspr.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --network=host --name=sspr-container -v /data/sspr:/config --stop-timeout 100 sspr/sspr-webapp:latest
```

Updating Containers on Distributed Servers

The containers must be updated in the following order:

- ♦ [“Updating Identity Manager Engine Container” on page 51](#)
- ♦ [“Updating Remote Loader Container” on page 52](#)
- ♦ [“Updating Fanout Agent Container” on page 53](#)
- ♦ [“Updating iManager Container” on page 53](#)
- ♦ [“Updating OSP Container” on page 54](#)
- ♦ [“Updating PostgreSQL Container” on page 55](#)
- ♦ [“Updating Identity Applications Container” on page 56](#)
- ♦ [“Updating Form Renderer Container” on page 56](#)
- ♦ [“Updating ActiveMQ Container” on page 57](#)
- ♦ [“Updating Identity Reporting Container” on page 57](#)
- ♦ [“Updating SSPR Container” on page 57](#)

Updating Identity Manager Engine Container

- 1 Create a `credentials.properties` file under the shared volume `/data` with the following content.

```
ID_VAULT_ADMIN="<ID_VAULT_ADMIN>"
ID_VAULT_PASSWORD="<ID_VAULT_PASSWORD>"
```

where, ID_VAULT_ADMIN must be in dot format.

For example,

```
ID_VAULT_ADMIN="admin.sa.system"
ID_VAULT_PASSWORD="novell"
```

- 2 From the location where you have extracted the Identity_Manager_4.8.2_Containers.tar.gz file, navigate to the Identity_Manager_4.8.2_Containers directory.

- 3 Run the following command to load the image:

```
docker load --input IDM_482_identityengine.tar.gz
```

- 4 Update the container using the following command if you are deploying the Identity Manager Engine using the overlay network:

```
docker run -d --ip=192.168.0.12 --network=idmoverlaynetwork --
hostname=identityengine.example.com --name=engine-container -v /etc/
hosts:/etc/hosts -v /data:/config -p 8028:8028 -p 524:524 -p 389:389 -p
8030:8030 -p 636:636 -e SILENT_INSTALL_FILE=/config/
credentials.properties --stop-timeout 100 identityengine:idm-4.8.2
```

Update the container using the following command if you are deploying the Identity Manager Engine using the host network:

```
docker run -d --network=host --name=engine-container -v /etc/hosts:/
etc/hosts -v /data:/config -e SILENT_INSTALL_FILE=/config/
credentials.properties --stop-timeout 100 identityengine:idm-4.8.2
```

Updating Remote Loader Container

- 1 From the location where you have extracted the Identity_Manager_4.8.2_Containers.tar.gz file, navigate to the Identity_Manager_4.8.2_Containers directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_remoteloader.tar.gz
```

- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.2 --network=idmoverlaynetwork --
hostname=remoteloader.example.com -p 8090:8090 --name=rl-container -v /
etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100
remoteloader:idm-4.8.2
```

The driver files can be found at the /opt/novell/eDirectory/lib/dirxml/classes/ directory of the container.

- 4 Start the Remote Loader instances.

Updating Fanout Agent Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_fanoutagent.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.3 --network=idmoverlaynetwork --hostname=fanoutagent.example.com --name=foa-container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 fanoutagent:idm-4.8.2
```
- 4 Start Fanout Agent.

Updating iManager Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_iManager323.tar.gz
```
- 3 Ensure that the `iManager.env` file is created and present in the `/data` directory.

```
# Certificate Public Key Algorithm
# Allowed Values: RSA, ECDSA256, ECDSA384
CERTIFICATE_ALGORITHM=RSA
# Cipher Suite
# Allowed Values:
# For RSA - NONE, LOW, MEDIUM HIGH
# For ECDSA256 - SUITEB128ONLY
# For ECDSA384 - SUITEB128, SUITEB192
CIPHER_SUITE=NONE
# Tomcat Server HTTP Port
TOMCAT_HTTP_PORT=8080
# Tomcat Server SSL Port
TOMCAT_SSL_PORT=8743
# iManager Authorized User (admin_name.container_name.tree_name)
AUTHORIZED_USER=
```
- 4 Update the container using the following command:

```
docker run -d --ip=192.168.0.4 --name=iman-container --network=idmoverlaynetwork --hostname=imanager.example.com -v /etc/hosts:/etc/hosts -v /data:/config -v /data/iManager.env:/etc/opt/novell/iManager/conf/iManager.env -p 8743:8743 --stop-timeout 100 imanager:3.2.3
```
- 5 (Conditional) If you have already installed Identity Manager, run the following command to check whether the plug-ins are loaded.

```
docker log <container name>
```

For example,

```
docker log <iman-container>
```

6 To install the Identity Manager plug-ins, perform the following steps:

6a Log in to iManager.

```
https://imanager.example.com:8743/nps/
```

6b Click **Configure**.

6c Click **Plug-in Installation** and then click **Available NetIQ Plug-in Modules**.

6d Select all the plug-ins from the **NetIQ Plug-in Modules** list and then click **Install**.

To obtain the plug-ins offline, perform the following steps:

1. Download the `Identity_Manager_4.8.2_Linux.iso` from the NetIQ Downloads website.
2. Mount the downloaded `.iso`.
3. From the mounted location, navigate to the `/iManager/plugins` directory and obtain the required plug-ins.

Alternatively, you can install the plug-ins from the [iManager plug-ins website](#).

7 Restart the iManager container.

```
docker restart iman-container
```

Updating OSP Container

1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

2 Run the following command to load the image:

```
docker load --input IDM_482_osp.tar.gz
```

3 Update the container using the following command:

```
docker run -d --ip=192.168.0.5 --network=idmoverlaynetwork --hostname=osp.example.com -p 8543:8543 --name=osp-container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 osp:idm-4.8.2
```

4 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it osp-container bash
```

5 Navigate to the `/opt/netiq/idm/apps/configupdate/` directory.

6 Modify the `configupdate.sh.properties` file.

7 Set the value of the `no_nam_oauth` parameter to `false`.

8 Save the `configupdate.sh.properties` file.

9 Run the following command to exit the container.

```
exit
```

Updating PostgreSQL Container

NOTE

- ♦ *(Conditional) This procedure applies only if you are updating PostgreSQL from the Identity Manager 4.8 version to the 4.8.2 version.*

In other words, do not update the PostgreSQL container if you have already deployed the PostgreSQL version (12.2) shipped with the Identity Manager 4.8.1 containers release. This release does not support any new version of the PostgreSQL container.

- ♦ Before you update the PostgreSQL container, ensure that you stop the dependent containers such as Identity Applications and/or Identity Reporting.
-

- 1 On the Docker host, navigate to any location. For example:

```
cd /tmp
```

- 2 Run the following command to take a back up of the existing PostgreSQL container data.

```
docker exec postgresql-container pg_dumpall -U postgres > dump.sql
```

- 3 Stop the PostgreSQL container.

```
docker stop <container name>
```

For example,

```
docker stop postgresql-container
```

- 4 Delete the PostgreSQL container.

```
docker rm <container name>
```

- 5 Delete the existing PostgreSQL data directory.

```
rm -rf /data/postgres
```

- 6 (Conditional) Delete the PostgreSQL Docker image.

```
docker rmi <image ID>
```

- 7 Create a sub-directory under the shared volume /data, for example, postgres.

```
mkdir postgres
```

- 8 From the location where you have extracted the Identity_Manager_4.8.2_Containers.tar.gz file, navigate to the Identity_Manager_4.8.2_Containers directory.

- 9 Run the following command to load the image:

```
docker load --input IDM_482_postgres.tar.gz
```

- 10 Update the container using the following command:

```
docker run -d --ip=192.168.0.6 --network=idmoverlaynetwork --hostname=postgresql.example.com --name=postgresql-container -p 5432:5432 -e POSTGRES_PASSWORD=<password> -v /data/postgres:/var/lib/postgresql/data -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 postgres:12.2-alpine
```

For example,

```
docker run -d --ip=192.168.0.6 --network=idmoverlaynetwork --
hostname=postgresql.example.com --name=postgresql-container -p
5432:5432 -e POSTGRES_PASSWORD=novell -v /data/postgres:/var/lib/
postgresql/data -v /etc/hosts:/etc/hosts -v /data:/config --stop-
timeout 100 postgres:12.2-alpine
```

- 11 Copy the data file you backed up on the Docker host ([Step 2](#)) to the new PostgreSQL data directory.

```
cp /tmp/dump.sql /data/postgres
```

- 12 Run the following command to log in to the container:

```
docker exec -it <container> <command>
```

For example,

```
docker exec -it postgresql-container bash
```

- 13 Navigate to the `/var/lib/postgresql/data/` directory.

- 14 Restore the data backed up in [Step 2](#) to the new PostgreSQL container.

```
psql -U postgres < dump.sql
```

- 15 Run the following command to exit the container.

```
exit
```

Updating Identity Applications Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_identityapplication.tar.gz
```

- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.7 --network=idmoverlaynetwork --
hostname=identityapps.example.com -p 18543:18543 --name=idapps-
container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100
identityapplication:idm-4.8.2
```

Updating Form Renderer Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.

- 2 Run the following command to load the image:

```
docker load --input IDM_482_formrenderer.tar.gz
```

- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.8 --network=idmoverlaynetwork --
hostname=formrenderer.example.com -p 8600:8600 --name=fr-container -v /
etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100
formrenderer:idm-4.8.2
```


Updating ActiveMQ Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_activemq.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.9 --network=idmoverlaynetwork --hostname=activemq.example.com -p 8161:8161 -p 61616:61616 --name=amq-container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 activemq:idm-4.8.2
```

Updating Identity Reporting Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_identityreporting.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.10 --network=idmoverlaynetwork --hostname=identityreporting.example.com -p 28543:28543 --name=rpt-container -v /etc/hosts:/etc/hosts -v /data:/config --stop-timeout 100 identityreporting:idm-4.8.2
```

Updating SSPR Container

- 1 From the location where you have extracted the `Identity_Manager_4.8.2_Containers.tar.gz` file, navigate to the `Identity_Manager_4.8.2_Containers` directory.
- 2 Run the following command to load the image:

```
docker load --input IDM_482_sspr.tar.gz
```
- 3 Update the container using the following command:

```
docker run -d --ip=192.168.0.11 --network=idmoverlaynetwork --hostname=sspr.example.com --name=sspr-container -v /etc/hosts:/etc/hosts -v /data/sspr:/config -p 8443:8443 --stop-timeout 100 sspr/sspr-webapp:latest
```


4 Troubleshooting

This section provides useful information for troubleshooting problems with the Identity Manager containers.

Identity Applications Container Displays Portlet Registration Exception

While deploying Identity Applications container, it displays the following exception:

```
ERROR
[com.novell.afw.portlet.consumer.core.EboPortletProducerChangeListener]
(main) [RBPM] Portlet registration with portletID: 'HeaderPortlet' does not
exist.
com.novell.afw.portlet.exception.EboPortletRegistrationException: Portlet
registration with portletID: 'HeaderPortlet' does not exist.
```

To workaround this issue, restart the Identity Applications container.

5 Best Practices

This section includes some tips and best practices for deploying Docker containers:

- ◆ NetIQ recommends you to set a limit on the amount of CPU used for a container. This can be achieved by using the `--cpuset-cpus` flag in the docker run command.
- ◆ To set a restart policy for a container, use the `--restart` flag in the docker run command. It is recommended to choose the on-failure restart policy and limit the restart attempts to 5.
- ◆ To set a limit on the memory used by a container, use the `--memory` flag in the docker run command.
- ◆ To gracefully stop a container, use the `--stop-timeout` flag. NetIQ recommends you to set the value of this flag to 100. If there are any active processes running inside the container, the container waits for 100 seconds and then exits. If all the processes are killed before the time specified in the `--stop-timeout` flag, the container exits when the last process is killed.
- ◆ To redirect the default log output to customized docker logs, use the `LOGTOFOLLOW` flag with the docker run command. For example, if you want to follow the new logs for OSP, specify the `-e LOGTOFOLLOW="<list of files separated by space>"` in the docker run command. This prints the logs in the new docker logs. You can use the `docker logs -f <container-name>` command to monitor the log files. The default logs for each containers are listed in the following table.

Container	Default logs
Identity Manager Engine	<code>/var/opt/novell/eDirectory/log/ndsd.log</code>
OSP	<code>/opt/netiq/idm/apps/tomcat/logs/catalina.out</code>
Identity Applications	<code>/opt/netiq/idm/apps/tomcat/logs/catalina.out</code>
Form Renderer	<code>/opt/netiq/idm/apps/sites/logs/formslogger.log</code>
ActiveMQ	<code>/opt/netiq/idm/activemq/data/activemq.log</code>
Identity Reporting	<code>/opt/netiq/idm/apps/tomcat/logs/catalina.out</code>

- ◆ For all containers except Remote Loader and Fanout Agent, you can monitor the health of the containers. Based on your requirement, you can customize the health status using the Docker runtime health checks. For example, to check the health of the `rdxml` service, use the `--health-cmd "ps -eaf | grep -i rdxml" --health-interval 60` flag.
- ◆ If you want to back up the trace files for the deployed drivers, then you can place the trace file under `/config/idm/` or manually copy the trace file to the volumized folder.

- ♦ To set a limit on the number of processes allowed to run at any point in time, use the `--pids-limit` flag in the `docker run` command. It is recommended to limit the PID value to 300.
- ♦ For Identity Manager Engine container, if you want to view the `environ` file located at the `/proc` directory of the `/proc` file system, use the `--cap-add=SYS_PTRACE` flag in the `docker run` command. By default, most of the privileges are restricted and only the required privileges are enabled. For more information, see [Docker](#) documentation.
- ♦ As a best practice, it is recommended to map individual data volume for each component.
- ♦ Ensure that the third party jar files are volume mounted so that they are available when the container is started every time. For example, if the `ojdbc.jar` is present in the `/opt/netiq/idm/apps/tomcat/lib` directory of the container, then you must volume mount the jar file using the following command:

```
-v /host/ojdbc.jar:/opt/netiq/idm/apps/tomcat/lib/ojdbc.jar
```

For example, run the following sample command containing all the above arguments for deploying containers:

```
docker run -d --name=<assign a name to the container> --network=<> --cap-add=SYS_PTRACE --pids-limit <tune container pids limit> --memory=<maximum amount of memory container can use> --restart=on-failure:5 --cpuset-cpus=<CPUs in which to allow execution> --network=<connect a container to network> --stop-timeout 100 -e LOGTOFOLLOW "/opt/netiq/idm/apps/tomcat/logs/catalina.out /opt/netiq/idm/apps/tomcat/logs/idapps.out" --health-cmd "ps -eaf | grep -i tomcat" --health-interval 60 -v <bind mount a volume> <image name>
```