Keys and Keystore

The Cisco IoT FND OVA includes keys and certificates stored in a keystore. Keys are cryptographic components used for encrypting and decrypting data, while certificates are digital documents that verify the authenticity of a device or entity within the network.

Here are the default values for the keys and certificates:

Environment	Keystore Location	Keystore Name	Default Password
Cisco IoT FND OVA Linux Host	`/opt/fnd/data/`	`cgms_keystore.selfsigned`	N/A
Cisco IoT FND container	`/opt/cgms/server/cgms/conf/`	`cgms_keystore`	`Public123!`

Note When you restart the Cisco IoT FND container, the values of /opt/cgms/server/cgms/conf/cgms_keystore file in Cisco IoT FND container is overwritten by /opt/fnd/data/cgms_keystore file. If /opt/fnd/data/cgms_keystore file is not present in the host, then the /opt/fnd/data/cgms_keystore.selfsigned file is used.

When you install Cisco IoT FND OVA for the first time, each certificate/key entry is referenced by an alias name in the keystore. The default alias are:

- cisco_sudi (cisco root CA certificate with 2029 expiry)
- jmarconi (cisco certificate)
- cgms (self signed certificate that is used by Cisco IoT FND when communicating with devices it has to manage)

The cgms certificate in the `/opt/cgms/server/cgms/conf/cgms_keystore` file on the Cisco IoT FND container and the `/opt/fnd/data/cgms_keystore.selfsigned` file on the Linux host initially contain a self-signed certificate from Cisco IoT FND. To create a custom cgms_keystore at `/opt/fnd/data` on the Linux host, you can either copy the existing `cgms_keystore.selfsigned` file or build it from scratch. The `/opt/fnd/data/cgms_keystore` file takes precedence if both files are present.

Note

- Ensure NTP is synchronized across the issuing Certificate Authority (CA) server, Cisco IoT FND, TPS, and FAR/HER, as it's crucial for Public Key Infrastructure.
- Change the hostname or IP address of the Cisco IoT FND host before issuing the certificate and building the cgms_keystore.
- The SAN field in the Cisco IoT FND certificate must include the hostname. If it includes the IP address, reissue the certificate if the hostname or IP changes.

Default Certificates and Keys in the cgms_keystore

Certificate/Key	Description
Issuing CA Certificate	The certificate of the organization's issuing CA server, which can be a root or intermediate CA. If intermediate, import both root and intermediate CA certificates into the keystore.
Cisco IoT FND Device Certificate	Issued for Cisco IoT FND by the organization's issuing CA server.
Cisco SUDI with 2029 Expiry	The Cisco manufacturer certificate for Cisco IoT FND, issued by Cisco, expiring in 2029.
Cisco SUDI with 2099 Expiry	The Cisco manufacturer certificate for Cisco IoT FND, issued by Cisco, expiring in 2099.

- Build cgms_keystore File, on page 2
- Encrypt and Change cgms_keystore Password, on page 4
- Manage Custom Web Certificates, on page 4
- Install Custom Browser Certificates, on page 5
- Use Keytool to Create the cgms_keystore File, on page 12
- Properties of Cisco IoT FND and TPS Configuration, on page 13

Build cgms_keystore File

Here are the instructions on how to build cgms_keystore file from the scratch that contains the required certificates and keys.

Procedure

Step 1 Change directory to /opt/fnd/data on linux host.

Example:

```
# cd /opt/fnd/data
```

Step 2 To create the keystore file with the keytool command, ensure it is named **cgms_keystore**, as Cisco IoT FND requires this specific name. First, copy your organization issuing CA certificate to a location on the server, such as /root/rootca.pem, using a file transfer method like SCP. The certificate can be in .cer, .crt, or .pem format. In this example, since the issuing CA is the root CA, use the alias name "root."

Example:

```
# keytool -import -trustcacerts -keystore /opt/fnd/data/cgms_keystore -alias
 root -file /root/rootca.pem
```

Convert the keystore from jks to pkcs12

```
# keytool -importkeystore -srckeystore /opt/fnd/data/cgms_keystore
-destkeystore /opt/fnd/data/cgms_keystore -deststoretype pkcs12
```

Verify that the file has been created by listing the contents of the keystore.

```
# keytool -list -keystore /opt/fnd/data/cgms_keystore
```

Step 3 Import a Cisco IoT FND certificate into your system. Use the following command to create a key pair and a Certificate Signing Request (CSR). The key pair is like a lock and key for your system, and the CSR is a request to get a digital certificate that matches this key.

Example:

```
keytool -genkeypair -keyalg RSA -keysize 2048 -alias cgms -ext "SAN=dns.labfnd.cisco.com, ip:1.0.0.1"
 -keystore /opt/fnd/data/cgms_keystore -dname "CN=labfnd, OU=iotescblr, O=cisco, L=Bengaluru,
ST=Karnataka, C=IN"
```

Note
The key size here is set to 2048, but you can use 4096 for extra security.

Step 4 Use the following command to create the CSR file required by the CA to issue your certificate.

Example:

```
keytool -certreq -file labfnd.csr -keystore /opt/fnd/data/cgms_keystore -alias cgms -ext
"SAN=dns:labfnd.cisco.com,ip:1.0.0.1"
```

Step 5 Send this CSR file to the CA. They will give you back a certificate for your server.

Step 6 Once you receive the certificate, for example, as a file named `labfnd.pem`, copy it to your server.

Step 7 Use the following command to import the certificates:

Example:

```
keytool -import -trustcacerts -keystore /opt/fnd/data/cgms_keystore -alias cgms -file /root/labfnd.pem
```

Step 8 If your Cisco IoT FND certificate is in a .pfx format, you need to know the alias name used in your .pfx file before importing it. Use the following command to find the alias name:

Example:

```
keytool -list -v -keystore /opt/fnd/data/nms.pfx -srcstoretype pkcs12 | grep Alias
```

Step 9 Once you have the alias name from the .pfx file, you need to import it into your keystore using the alias cgms.

Example:

```
keytool -importkeystore -v -srckeystore /opt/fnd/data/nms.pfx -destkeystore /opt/fnd/data/cgms_keystore
 -srcalias le-IoT FND-8f0908aa-dc8d-4101-a526-93b4eaad9481 -destalias cgms
```

You've built the cgms_keystore file.

What to do next

Import SUDI with 2029 and 2000 expiries.

Encrypt and Change cgms_keystore Password

Ensure the `cgms.properties` file contains the encrypted password for `cgms_keystore` so the Cisco IoT FND application can access it. When you create the `cgms_keystore` for the first time, encrypt its password and include this encrypted password in the `cgms.properties` file. If you ever change the password for `cgms_keystore`, encrypt the new password and update it in the `cgms.properties` file.

Use the following instructions to encrypt and set a new password:

Procedure

Step 1 Run the following command to encrypt the password for the new **cgms_keystore**.

Example:

```
# docker exec -it fnd-container /opt/cgms/bin/encryption_util.sh
encrypt <keystore password>
# docker exec -it fnd-container /opt/cgms/bin/encryption_util.sh
encrypt cisco123
#2bVvZsq+vsq94YxuAKdaag--
```

Step 2 Modify the **cgms.properties** file in the /opt/fnd/data folder, and edit the following line to set the new encrypted **cgms_keystore** password:

Example:

```
cgms-keystore-password-hidden=<encrypted new cgms_keystore password>
```

You've now successfully encrypted and generated a new cgms_keystore password.

Manage Custom Web Certificates

Custom web certificates are digital certificates that are specifically created and configured for a particular organization or application rather than using standard, off-the-shelf certificates.

The web certificate details are not retained after a reboot of Cisco IoT FND. You should perform a back up of the following files before you attempt to reboot Cisco IoT FND.

Procedure

Step 1 Use SSH or direct console access to log into the server where Cisco IoT FND is running.

Step 2 Open a terminal and navigate to the `/opt/cgms/server/cgms/conf/` directory.

Example:

```
cd /opt/cgms/server/cgms/conf/
```

Step 3 Decide on a location for your backup files. It could be another directory on the same server or an external storage location.

Example:

```
mkdir -p /path/to/backup_directory
```

Step 4 Use the cp command to copy each file to the backup directory. Replace /path/to/backup_directory with your actual backup directory path:

Example:

```
cp jbossas.keystore.password /path/to/backup_directory/
cp jbossas.keystore /path/to/backup_directory/
cp VAULT.dat /path/to/backup_directory/
cp vault.keystore /path/to/backup_directory/
cp standalone.xml /path/to/backup_directory/
cp cgms.conf /path/to/backup_directory/
```

Step 5 Ensure all files are copied correctly by listing the contents of the backup directory:

Example:

```
ls -l /path/to/backup_directory/
```

Step 6 Reboot the server for the changes to take effect.

What to do next

Install Custom Browser Certificates.

Install Custom Browser Certificates

By default, Cisco IoT FND installations use a self-signed certificate for HTTP(S) communication with either a client web browser or the NB API client. If necessary, you can use certificates signed by your CA servers. This section provides installation procedures for these custom certificates.

Use the following instructions to install custom certificates in the browser client. The instructions assume that the Java environment has been set to use the Java bundled with Cisco IoT FND in /opt/cgms/jre.

Note Update the jbossas.keystore in Cisco IoT FND container. The keystore is present in: /opt/cgms/server/cgms/conf/. The name of the keystore is jbossas.keystore.

Before you begin

- Clear the client browser cache.
- Remove existing certificates for the NMS server (by IP and DNS) on the client browser

  In Firefox, for example, select **Preferences** > **Advanced** > **Encryption** > **View Certifications**. Remove the certificates in the list for the respective server.
- Choose a common name to use in the signed certificate.

  This name requires a DNS entry that resolves to the NMS server IP address.
- Generate the new certificates and export them to a.PFX file.

  This file must contain the private keys, public certificate, and CA server certificates.

See Use Keytool to Create the cgms_keystore File for the procedure to generate the private and public keys for the cgms_keystore file and export them to a .PFX file.

Procedure

Step 1 Copy the certificate and private key *.pfx file from the host to container.

Example:

```
docker cp newcert.pfx fnd-container:/opt/cgms/server/cgms/conf/
```

Step 2 Run the following command to enter the container:

Example:

```
docker exec -i -t fnd-container /bin/bash
```

Step 3 Change directory to /opt/cgms/server/cgms/conf/ in the container:

Example:

```
cd /opt/cgms/server/cgms/conf/
```

Step 4 On the NMS server, copy the existing jbossas.keystore, jbossas.keystore.password, vault.keystore, and VAULT.dat files from the /opt/cgms/server/cgms/conf/ directory to a safe location.

Step 5 Delete the existing jbossas.keystore, jbossas.keystore.password, vault.keystore, and VAULT.dat files from the /opt/cgms/server/cgms/conf/ directory.

Step 6 Determine the alias in the .PFX file that you plan to import into the new jbossas.keystore file:

Example:

```
#keytool -list -v -keystore newcert.pfx -storetype pkcs12
Enter the keystore password: keystore_password_when_pfx_file_was_created
Keystore type: PKCS12
Keystore provider: SunJSSE
Your keystore contains 1 entry
Alias name: le-nms-a88ef13a-a519-457f-a2e1-0540f5453ee0
Creation date: Feb 23, 2018
Entry type: PrivateKeyEntry
Certificate chain length: 2
Certificate[1]:
…
```

Step 7 Import the new custom certificate, in .PFX file format into a new jbossas.keystore file; and, at the same time change the alias name to **jboss**. Follow the prompts:

Example:

```
# keytool -importkeystore -v -srckeystore newcert.pfx -srcstoretype pkcs12
-destkeystore /opt/cgms/server/cgms/conf/jbossas.keystore -deststoretype jks
-srcalias le-nms-a88ef13a-a519-457f-a2e1-0540f5453ee0 -destalias
jboss -destkeypass your_keystore_password
Enter destination keystore password: your_keystore_password
Enter source keystore password: keystore_password_when_pfx_file_was_created
[Storing /opt/cgms/server/cgms/conf/jbossas.keystore]
```

Step 8 The keystore password is stored in the /opt/cgms/server/cgms/conf/VAULT.dat file. Perform the following steps to update the keystore password to match the keystore password.

Step 9 Create a new vault.keystore file:

Example:

```
/opt/cgms/jre/bin/keytool -genseckey -alias vault -storetype jceks
-keyalg AES -keysize 256 -storepass keystore -dname "CN=IoTFND, OU=IoT, O=Cisco Systems,
L=San Jose, ST=CA, C=US" -keypass keystore -validity 730 -keystore vault.keystore
```

Step 10 Update the VAULT.dat file with the new password:

Example:

```
/opt/cgms/bin/vault.sh --keystore /opt/cgms/server/cgms/conf/vault.keystore
 --keystore-password keystore --alias vault --vault-block keystore_pass
--attribute password --sec-attr keystore --enc-dir /opt/cgms/server/cgms/conf/
 --iteration 50 --salt 12345678 -n

=========================================================

  JBoss Vault

  JBOSS_HOME: /opt/cgms

  JAVA: /opt/cgms/jre/bin/java

========================================================================

WFLYSEC0047: Secured attribute value has been stored in Vault.
Please make note of the following:
********************************************
Vault Block:keystore_pass
Attribute Name:password
Configuration should be done as follows:
VAULT::keystore_pass::password::1
********************************************
WFLYSEC0048: Vault Configuration commands in WildFly for CLI:
********************************************
For standalone mode:
/core-service=vault:add(vault-options=[("KEYSTORE_URL" =>
"/opt/cgms/server/cgms/conf/vault.keystore"),
("KEYSTORE_PASSWORD" => "MASK-0VKsAwH928fwt.3H2qUwOG"),("KEYSTORE_ALIAS" => "vault"),
("SALT" => "12345678"),("ITERATION_COUNT" => "50"),("ENC_FILE_DIR" =>
"/opt/cgms/server/cgms/conf/")])
********************************************
For domain mode:
/host=the_host/core-service=vault:add(vault-options=[("KEYSTORE_URL" =>
"/opt/cgms/server/cgms/conf/vault.keystore"),
("KEYSTORE_PASSWORD" => "MASK-0VKsAwH928fwt.3H2qUwOG"),("KEYSTORE_ALIAS" => "vault"),
("SALT" => "12345678"),("ITERATION_COUNT" => "50"),("ENC_FILE_DIR" =>
 "/opt/cgms/server/cgms/conf/")])
********************************************
```

Step 11 Copy /opt/cgms/standalone/configuration/standalone.xml and standalone-cluster.xml to a safe location.

Step 12 Update the /opt/cgms/standalone/configuration/standalone.xml or standalone-cluster.xml file. Depending on whether the Cisco IoT FND server is standalone or clustered, update the respective file accordingly.

Step 13 Replace the keystore password:

Example:

```
<vault>
<vault-option name="KEYSTORE_URL"
value="/opt/cgms/server/cgms/conf/vault.keystore"/>

<vault-option name="KEYSTORE_PASSWORD" value="MASK-VKsAwH928fwt.3H2qUwOG"/>
<vault-option name="KEYSTORE_ALIAS" value="vault"/>
<vault-option name="SALT" value="12345678"/>
<vault-option name="ITERATION_COUNT" value="50"/>
```

```
<vault-option name="ENC_FILE_DIR" value="/opt/cgms/server/cgms/conf/"/>
</vault>
```

Step 14 Edit the standalone.xml or standalone-cluster.xml and replace the existing <vault> section with the above. Save and exit.

Step 15 Run the following command to restart the CGMS service:

Example:

```
/etc/init.d/cgms restart
```

Step 16 Use your browser to connect to the NMS server.

Step 17 Accept and add the new certificates.

Step 18 Use your browser to log in to Cisco IoT FND.

When you restart the container, it replaces the custom web certificates with self-signed certificates. It is recommended to restart or upgrade the container before taking a backup of the following files to the local host:

- jbossas.keystore
- jbossas.keystore.password
- VAULT.dat
- vault.keystore
- standalone.xml
- standalone-cluster.xml

Install custom certificates in the browser client for Cisco IoT FND Release 5.1 and later releases

This task helps you install custom certificates for a fresh install of Cisco IoT FND running Cisco IoT FND Release 5.1 and later releases.

Procedure

Step 1 Copy the certificate and private key *.pfx file from the host to the container:

Example:

```
docker cp custom_ca_cert.pfx fnd-container:/opt/cgms/server/cgms/conf/
```

a) Run `docker exec -i -t fnd-container /bin/bash` to enter the container.
b) Change the directory to `cd /opt/cgms/server/cgms/conf/`
c) On the NMS server, copy the existing `jbossas.keystore`, `jbossas.keystore.password`, `vault.keystore`, and `VAULT.dat files` from the `/opt/cgms/server/cgms/conf/` directory to a safe location.

Step 2 Determine the alias in the .PFX file that you plan to import into the new jbossas.keystore file:

Example:

```
#keytool -list -v -keystore custom_ca_cert.pfx -storetype pkcs12
```

Step 3 Enter the keystore password that you set when .PFX file was created

Example:

```
Enter the keystore password: keystore_password_when_pfx_file_was_created

Keystore type: PKCS12
Keystore provider: SunJSSE
Your keystore contains 1 entry
Alias name: lab5-win-2mfnersdbr3-ca
Creation date: Feb 23, 2025
Entry type: PrivateKeyEntry
Certificate chain length: 2
Certificate[1]:
…
```

a) Delete the existing jbossas.keystore from the `/opt/cgms/server/cgms/conf/` directory.

b) Import the new custom certificate, in .PFX file format into a new jbossas.keystore file; and, at the same time change the alias name to jboss. Follow the prompts:

Example:

```
keytool -importkeystore -v -srckeystore custom_ca_cert.pfx -srcstoretype pkcs12
-destkeystore /opt/cgms/server/cgms/conf/jbossas.keystore -deststoretype jks
-srcalias Alias name -destalias jboss -destkeypass your_keystore_password
```

The keystore password is stored in the `/opt/cgms/server/cgms/conf/VAULT.dat` file.

Step 4 Delete the following files:

a) Delete the **`vault.keystore`** file.

b) Delete the **`vault.dat`** file.

c) Create a new **`vault.keystore`** file:

Example:

```
/opt/cgms/jre/bin/keytool -genseckey -alias vault -storetype jceks
-keyalg AES -keysize 256 -storepass your_keystore_password -dname your_domain_name -keypass
-validity 730 -keystore vault.keystore
```

d) Update the vault.dat file with the new password:

Example:

```
/opt/cgms/bin/vault.sh --keystore /opt/cgms/server/cgms/conf/vault.keystore
 --keystore-password your_keystore_password --alias vault --vault-block keystore_pass
--attribute password --sec-attr your_keystore_password --enc-dir /opt/cgms/server/cgms/conf/
--iteration 50 --salt 12345678 -n
```

The vault.keystore contains the reference to vault.dat and vault.dat stores and hides the jboss keystore password. This command creates a new vault.dat file that contains the new jboss.keytsore password.

e) Copy `/opt/cgms/standalone/configuration/standalone.xml` to a safe location.

Step 5 Update the `/opt/cgms/standalone/configuration/standalone.xml`.

a) Replace the keystore password:

Example:

```
<vault>
<vault-option name="KEYSTORE_URL"
value="/opt/cgms/server/cgms/conf/vault.keystore"/>
<vault-option name="KEYSTORE_PASSWORD" value="MASK-0VKsAwH928fwt.3H2qUwOG"/>
```

```
<vault-option name="KEYSTORE_ALIAS" value="vault"/>
<vault-option name="SALT" value="12345678"/>
<vault-option name="ITERATION_COUNT" value="50"/>
<vault-option name="ENC_FILE_DIR" value="/opt/cgms/server/cgms/conf/"/>
</vault>
```

b) Edit `standalone.xml` and replace the existing section with the above.
c) Restart the CGMS service using `/etc/init.d/cgms/restart`.
d) Save and exit.

Step 6 Perform the following steps to retain the certificates upon container reloads.

a) Copy the files from `opt/cgms/server/cgms/conf/` to `/opt/fnd/data`

Note
If you see any of the following files in `/opt/fnd/data`, delete the files and then copy the files from the container using the following commands:

Example:

```
docker cp fnd-container: /opt/cgms/server/cgms/conf/jbossas.keystore /opt/fnd/data
docker cp fnd-container: /opt/cgms/server/cgms/conf/jbossas.keystore.password /opt/fnd/data
docker cp fnd-container: /opt/cgms/server/cgms/conf/vault.dat /opt/fnd/data
docker cp fnd-container: /opt/cgms/server/cgms/conf/vault.keystore /opt/fnd/data
docker cp fnd-container: /opt/cgms/standalone/configuration/stanalone.xml /opt/fnd/data
```

Step 7 Navigate to `/opt/fnd/scripts` and run the **`script upgrade.sh`** and enter option 2 **`custom web certificates`**

Step 8 Choose option 2 **`continer reload`** to reload the container

Step 9 Use your browser to connect to the NMS server.

a) Accept and add the new certificates.
b) Login to Cisco IoT FND.
c) Backup the following files in your local

- jbossas.keystore
- jbossas.keystore.password
- VAULT.dat
- vault.keystore
- standalone.xml

You've succesfully installed the custom certificates in the browser client for Cisco IoT FND Release 5.1 and later releases.

Retain custom certificates during Cisco IoT FND upgrades

This task helps you retain custom certificates from an older Cisco IoT FND version to the latest Cisco IoT FND version.

The `upgrade.sh` script needs the custom web certificate keystore (`jbossas.keystore`) in `/opt/fnd/data`. You must copy the `jbossas.keystore` to `/opt/fnd/data` so the OVA upgrade can locate and import the custom certificate.

Before you begin

- Ensure that you have custom certificates installed in your Cisco IoT FND. For more information see, Install custom browser certificates.
- Ensure that you are following the steps provided in the Upgrade Cisco IoT FND on OVA to perform an upgrade of your Cisco IoT FND.
- Ensure to login as a root user.

Procedure

Step 1 Get the upgrade package from Cisco and follow the steps in Upgrade Cisco IoT FND on OVA to upgrade your Cisco IoT FND.

Step 2 Run the RPM file.

Example:

```
http://10.78.81.200/fnd-postgres-ova/5.1.0/152/archive/upgrade-ova-5.1.0-152.rpm
rpm -Uvh --force upgrade-ova-5.1.0-152.rpm
```

The RPM verifies, prepares, and begins updating/installing.

Step 3 You change to `/opt/fnd/scripts/` and execute **`./upgrade.sh`**

Example:

```
Select the Type of Web certificate currently used
Option 1: FND Self Signed Certificate
Option 2: Custom Web Certificate
```

Step 4 Select **`Option 2: Custom Web Certificate`**

Example:

```
Enter option (1 or 2): 2
You selected Option 2 (Custom Web Certificate).
```

Step 5 The script reports a missing custom certificate keystore:

Example:

```
jbossas.keystore was not found in /opt/fnd/data.
Copy jbossas.keystore from /tmp to /opt/fnd/data and re-run the upgrade.
```

Step 6 Copy the `jbossas.keystore` from `/tmp/jbossas.keystore` to `/opt/fnd/data`

Example:

```
cp /tmp/jbossas.keystore /opt/fnd/data
```

Step 7 Re-run the **`./upgrade.sh`** script.

You've copied the custom certificates from a temporary directory to `/opt/fnd/data` which results in retaining the custom certificates to the upgraded Cisco IoT FND version.

Use Keytool to Create the cgms_keystore File

Use the instructions provided in this section to create the cgms_keystore file for both Cisco IoT FND and the TPS proxy.

Procedure

Step 1 Enter the following command on the Cisco IoT FND or TPS proxy server as root to view the contents of the .PFX file.

Example:

```
[root@ tps_server ~]# keytool -list -v -keystore nms_cert.pfx -srcstoretype pkcs12
```

Note
You can get the alias name from the .pfx file during import.

Step 2 Enter the keystore password when prompted. The password is the same as the one you used to create the .pfx file.

Step 3 Enter the following command to import the certificates into the cgms_keystore file:

Example:

```
keytool -importkeystore -v -srckeystore
filename.pfx -srcstoretype pkcs12
-destkeystore cgms_keystore -deststoretype jks -srcalias alias_name
-destalias cgms
-destkeypass
keystore_password
```

Step 4 Enter the destination keystore password when prompted.

Step 5 Re-enter the keystore password when prompted.

Step 6 Enter the password you used earlier while creating the .pfx file (either *nms_cert.pfx* or *tps_cert.pfx*) when prompted for the source keystore password.

Example:

```
# keytool -list -v -keystore nms_cert.pfx -srcstoretype pkcs12
Enter keystore password: keystore
Keystore type: PKCS12
Keystone provider: SunJSSE
Your keystore contains 1 entry
Alias name: le-cgnms-75edd1e3-7e65-41b4-97f1-a913ebf21c8b
Creation date: March 29.2018
Entry type: PrivateKey Entry
Certificate chain length: 3
Certificate[1]:
Owner: CN=nms.sgbu.cisco.com
Issuer: CN=cisco-RSA-SUBCA-CA, DC=cisco, DC=com
```

Note
This example diplays the steps for the *nms_cert.pfx*. To view the details on the *tps_cert.pfx* and import the certificates to the TPS proxy, use the same commands but replace the references to *nms_cert.pfx* with *tps_cert.pfx*, and use the alias name from the *tps_cert.pfx* file.

Example:

```
# keytool -importkeystore -v
-srckeystore nms_cert.pfx -srcstoretype pkcs12
-destkeystore cgms_keystore -deststoretype jks -srcalias
le-cgnms-75edd1e3-7e65-41b4-97f1-a913ebf21c8b -destalias cgms
-destkeypass
keystore_password

Enter destination keystore password: keystore_password
Re-enter new password: keystore_password
Enter source keystore password: keystore
...Storing cgms_keystore
```

The **storing** *cgms* **_keystore** text indicates successful completion.

Properties of Cisco IoT FND and TPS Configuration

The Cisco Gateway Management System (CGMS) and TPSPROXY properties in Cisco IoT FND and TPS are used to customize, optimize, and secure the management and transmission of gateway and telemetry data, ensuring effective network operation and performance.

CGMS Properties

Property Name	Example Value	Description
cgms-keystore-password-hidden=	< encrypted >	Encrypted password for the cgms keystore. Encrypt or decrypt with encryption_util.sh.
hsm-keystore-name=	testGroup1	HSM partition name.
hsm-keystore-password=	< encrypted >	Encrypted HSM partition password.
security-module=	ssm/hsm	Type of security module being used.
ssm-host=	<ipv4 address >	IP Address of SSM server.
ssm-port=	8445	Port of SSM server.
ssm-keystore-alias=	ssm_csmp	Alias name for SSM certificate in keystore.
ssm-keystore-password=	< encrypted >	Encrypted password for the SSM keystore.
ssm-key-password=	< encrypted >	Encrypted key for the SSM key.
multicast-interface-address=	< ipv6 address >	Cisco IoT FND IPv6 source address for multicast traffic.
dhcpV4ClientListenAddresses=	<ipv4 address >	IPv4 address on your Cisco IoT FND server used to exchange DHCPv4 messages.
dhcpV6ClientListenAddresses=	< ipv6 address >	IPv6 address on your Cisco IoT FND server used to exchange DHCPv6 messages.
OptimizeTunnelProv=	true/false	Indicates whether or not to lock the HER during tunnel provisioning.

Property Name	Example Value	Description
allowed-outage-skew=	5000	Allow outage skew in seconds, for outage or restoration events.
rf.validate-firmware-tlvs=	true/false	Skips CG-Mesh device firmware validation.
googleMapsClientId=	< Client ID >	Google maps client ID.
googleMapsApiKey=	< API key >	Google maps API key.
enable-bootstrap-service=	true/false	Used to enable PNP bootstrapping service.
scep-url=	http(s) :// < url of SCEP server >	URL of SCEP server.
ca-fingerprint=	< fingerprint of CA certificate >	Fingerprint of CA certificate.
proxy-bootstrap-ip=	<ipv4/v6 address or FQDN >	PNP server identity sent by Cisco IoT FND to the PNP agent.
bootstrap-fnd-alias=	subca	Alias name assigned to the CA certificate from the issuer in the Cisco IoT FND keystore.
pnp-server-port=	9125	PNP server port, default is 9125.
pnp-install-trustpool=	true/false	Send the CA bundle file which includes well known public CA certificates.
reload-during-bootstrap=	true/false	Indicates whether or not to reload a device after PNP boostrapping.
router-file-upload-retries	0	Number of retries for router file upload job
router-firmware-upload-retries	0	Number of retries for the firmware upload job.
router-firmware-install-retries	0	Number of retries for the firmware install job.
collect-cellular-link-metrics	true/false	Indicates whether or not to collect cellular metrics.
collect-cellular-link-metrics-interval	30	Interval for cellular metrics.
router-firmware-upload-timeout-minutes=	30	Firmware upload job timeout duration in minutes.
router-firmware-install-timeout-minutes=	60	Firmware install job timeout duration in minutes.
cgr-ha-fetch-mesh-key-attempts	3	Number of attempts to fetch the mesh keys.
cgr-ha-fetch-mesh-key-delay-mins	1	Number of minutes or interval between mesh-key-attempts.

TPSPROXY Properties

Property Name	Example Value	Description
cgdm-tpsproxy-addr=	<ipv4/v6 address or FQDN >	Source IP address of messages coming from the TPSProxy.
cgdm-tpsproxy-subject=	CN="common_name", OU="organizational_unit", O="organization", L="location", ST="state", C="country"	The exact certificate subject contained in the TPSPROXY's certificate.
bootstrap-proxy-listen-port=	9125	Port on which TPS is listening for HTTP traffic.
inbound-bsproxy-destination=	<ipv4/v6 address or FQDN >	IP address and port to forward info received from the router over HTTP.
outbound-proxy-allowed-addresses=	<ipv4/v6 address or FQDN >	Comma separated list of FQDN/IP addresses, the proxy allows outbound messages to originate from it.