**GMU Utility**

The Gateway Migration Utility (GMU) is a standalone, Java-based command-line tool for migrating supported entities and configurations between Gateways. GMU commands let you control which entities are migrated, and how they are migrated. GMU makes it easier to script migrations using the REST-based service, REST Management API.

Use GMU and the [REST Management API](https://docops.ca.com/ca-api-gateway/9-1/en/apis-toolkits-and-sdks/rest-management-api)to do these common tasks:

* Provision new CA API Gateway(for example, from Test to Production)
* Migrate policies between Gateways
* Create migration templates to:
  + Easily make minor changes to Gateways
  + Migrate the same bundle with different environment configurations to other Gateways

1. **Configure RESTMAN API and GMU utility**

**Step 1**

* Publish the RESTMAN API on the source and target gateways
  + 1. Start Policy Manager, Select **Tasks > Publish Internal Service**
    2. **Select Gateway REST Management Service and accept default.**
    3. **Click Finish and verify.**
    4. **Repeat the above steps at Target Gateway also.**

**Step 2**

* Create separate migration Admin Users as migration Admin users are authenticated using certs.

1. Login to Policy manager
2. Go to: **Tasks > Users and Authentication > Create Internal User** and create the migration Administrator user.
3. Click the **Identity Providers** tab.
4. Right-click **Internal Identity Providers**and select **Search Identity Provider**.
5. Double-click the migration Administrator in the list.
6. Click the **Roles** tab, and click **Add**.
7. Select the Administrator checkbox, and click **Add**.
8. Repeat steps 1-7 on any target CA API Gateway that is involved in migration.

**Step 3**

* Generate GMU Client certificate and Private Key

1. Go to **Tasks** **> Certificates, Keys, and Secrets >** **Manage Private Keys**.
2. Click **Create**.
3. Enter an alias and click **Create**. (**Note:** The alias must match the migration Administrator that you created in the previous step)
4. Double-click the private key in the list.
5. Click **Export Key**, optionally specify a password, and save the certificate and private key in the pkcs12 file
6. Move the pkcs12 file into a folder for use with GMU commands (for example, **--clientCert gmu.p12**).
7. Click **View Certificate**, click **Export**, and save the certificate file.
8. Record the location of the certificate file (.pem format). (You need it in the next step to add the certificate to the Internal identity Providers.)
9. (Optional). Delete the private key; it is not needed on the CA API Gateway.

**Step 4**

* Map Migration Administrators to the GMU Client certs

1. On the source CA API Gateway, log in to Policy Manager, click the **Identity Providers** tab.
2. Under **Identity Providers**, right-click **Internal Identity Provider** and select **Search Identity Provider**.
3. Double-click the migration Administrator.
4. Select the **Certificate** tab, click **Import**.
5. In the **Add Certificate** Wizard, select **Import from a File**, and import the certificate file (created in step 8 in the previous step).
6. Click **Browse** to find your certificate and complete the wizard.
7. Repeat steps 1-6 on target CA API Gateways.

**Step 5**

* Configure GMU Tools

1. Log in to: [CA Support](https://support.ca.com/).
2. In the **Download Center**, select **Download Products**.
3. Select your Gateway version from the list that appears and then complete the Product Downloads page.
4. Download the "Installers" DVD image. Extract the image to a directory.
5. Locate the file *GatewayMigrationUtility-x.x.xx* and extract its contents.
6. Configure Java environment variables.  
   **Windows**: Java SDK/JRE bin directory must be set on the PATH or JAVA\_HOME environment variable, or specified using the --jdk argument.  
   **Linux/Unix**: Java SDK/JRE bin directory must be set on the PATH, JAVA\_HOME, or SSG\_JAVA\_HOME environment variable, or specified using the --jdk argument.

**Step 6**

* Create a folder name “Certs” in your GMU folder created on your desktop. When we will save the .cer /.p12 / .pem , we need to double click and add it to the trust store.
* Establish the GMU server trust with the migration Gateways

1. On the source Gateway, log in to Policy Manager.
2. Go to: **Tasks > Certificates, Keys, and Secrets > Manage Private Keys**.
3. Double-click the default **ssl** private key ("S" in the last column).
4. Click **View Certificate**.
5. On the **General** tab, click **Export,**and save the file.
6. On the computer running GMU, verify that JAVA\_HOME is configured.
7. Using the Java keytool, add the certificate to the JDK trust store. Commnad :

*$JAVA\_HOME/bin/keytool -importcert -alias <ssgcert\_alias> -file <path to certificate file> -keystore $JAVA\_HOME/jre/lib/security/cacerts -storepass <trust\_store\_password>*

1. Repeat steps 1-7 on target Gateways.

* Run GMU : As I tried in Windows so tested with: Run 🡪Start -- cmd – GatewayMigration.bat
* If you are checing in Linux, then we need to test it via ./GatewayMigrationUtility.sh

**Example:**

* A complete scenario where we are going to migrate the APIs from lower to higher environment. Below minimal steps we can follow in order to migrate the complete setup (including JDBC setup, cluster-wide properties etc)

We will follow the below mentioned steps one by one in order to migrate the APIs

1.) I have created the clientcertificate in the “Step 3” mentioned above. I have used the password “L7Secure$0@” and now I need to encrypt the password by running the below command:

*GatewayMigrationUtility.bat encodePassword --password L7Secure$0@ > mypass.txt*

2.) Once we encoded the password, it will get save under GMU folder at my Desktop with name “mypass.txt” with path: *C:\Users\SINPRAT\Desktop\GMU\GatewayMigrationUtility-1.3.00\mypass.txt*

3.) Now we need to migrate the required policy i.e. from DEV environment. To achieve this we need to use migrateOut command:

*GatewayMigrationUtility.bat migrateOut --host 192.168.108.128 --clientCert dev.p12 --password GlPDkJPcmscJTtqM2wTOsA --dest dev.xml --defaultAction NewOrUpdate --trustCertificate --trustHostname --service eb8b46d86703a9c224d30de1d6d44155 --folderName /Dev*

**Where**;

--hostName: Name of the host from which you want to migrate (In our case it is DEV environment)

--dest: It is an xml file is ot be created which contains policies in xml format with all the details including cluster wide properties and jdbc connections

4.) We have to create template properties for the required policy (dev.xml) by templating command:

*GatewayMigrationUtility.bat template --bundle dev.xml --template devtemplate.properties*

5.) Now we can change the values which are different than lower environment. Once the value got changed, we will save the file, then we will create xml file again after doing detemplate in next step.

6.) “De-template” into xml file from modified “devtemplate” template properties file:

*GatewayMigrationUtility.bat detemplate --bundle dev.xml --template devtemplate.properties*

Now it will create xml file dev.xml with modified values under GMU folder saved in my local Desktop.

5.) “Migrate-in” the policy in higher environment

*GatewayMigrationUtility.bat migrateIn --host 192.168.108.129 –clientCert test.p12 --password GlPDkJPcmscJTtqM2wTOsA --bundle dev.xml --results results.xml --destFolder /Test/*

--bundle: Modified XML file

--destFolder: Destination folder where we need to migrate the policies in higher environment

--results: It creates an XML file which shows the results of migration whether migration is successful or failed with proper reason.

1. **What GMU can do and what can’t:**

***Can:***

* Provision new CA API Gateway
* Migrate policies between Gateways
* GMU commands let you control which entities are migrated, and how they are migrated
* GMU migrates encapsulated assertions and you can securely export/import passwords during migration.
* We can use GMU for backing up policies and services and other CA API Gateway entities.
* We can migrate the dependencies too.

***CanNot***

* The GMU cannot migrate system data (OS-level files and database), such as system.properties and host.properties. For migrating OS level files we need another utility tool name: ssgmigrate.sh
* We cannot roll back to previous migration version of a Gateway