

Oracle® Cloud

Migrate to Oracle Mobile Hub



Release 20.2.3
F15109-06
August 2020

The Oracle logo, consisting of the word "ORACLE" in white, uppercase, sans-serif font, centered within a solid red square.

ORACLE®

Oracle Cloud Migrate to Oracle Mobile Hub, Release 20.2.3

F15109-06

Copyright © 2019, 2020, Oracle and/or its affiliates.

Primary Author: Catherine Pickersgill

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

Contents

Preface

| | |
|-----------------------------|---|
| Audience | v |
| Documentation Accessibility | v |
| Conventions | v |

1 About Migrating to Oracle Mobile Hub

| | |
|-------------------------------|-----|
| Learn About Oracle Mobile Hub | 1-1 |
| Plan your Migration Strategy | 1-2 |
| Data Types | 1-3 |
| Roles | 1-3 |

2 The Migration Tool

| | |
|--------------------------------------|------|
| Patch OMCe | 2-1 |
| Edit the Patch Script | 2-2 |
| Save the Patch Script to the Core VM | 2-3 |
| Run the Patch Script | 2-3 |
| Restart the Core VM | 2-5 |
| Troubleshoot Patch Issues | 2-5 |
| About the Configuration File | 2-5 |
| Configure config.json for MCS | 2-7 |
| Migration Scope | 2-8 |
| Source Instance | 2-9 |
| Target Instance | 2-9 |
| Storage Account | 2-10 |
| Security | 2-11 |
| Configure config.json for OMCe | 2-11 |
| Migration Scope | 2-11 |
| Source Instance | 2-12 |
| Target Instance | 2-13 |
| Storage Account | 2-13 |

3 Migrate to Oracle Mobile Hub

| | |
|---|-----|
| Build the Docker Container | 3-1 |
| Test the Migration | 3-1 |
| Migration Tool Commands | 3-2 |
| Perform the Migration | 3-4 |
| Use a Custom config.json | 3-4 |
| User-Isolated Collection Objects (MCS Only) | 3-4 |

4 Security Configuration (MCS Only)

| | |
|---|-----|
| Migrate Users | 4-1 |
| Export users from Traditional Cloud Accounts | 4-1 |
| Edit the Users File | 4-1 |
| Import Users into Oracle Identity Cloud Service | 4-2 |

Preface

Welcome to Oracle Mobile Hub.

Audience

This guide is intended for service administrators responsible for migrating mobile instances to Mobile Hub.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Conventions

The following text conventions are used in this guide:

| Convention | Meaning |
|-----------------|--|
| boldface | Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary. |
| <i>italic</i> | Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values. |
| monospace | Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter. |

1

About Migrating to Oracle Mobile Hub

Learn about the benefits of migrating your existing instances from Oracle Mobile Cloud Service or Oracle Mobile Cloud, Enterprise to Oracle Mobile Hub, and understand the migration process and tools.

New features are being introduced into Mobile Hub and if you want to use those features you will need to migrate your mobile cloud configurations from Oracle Mobile Cloud Service (MCS) or Oracle Mobile Cloud Enterprise (OMCe) to Mobile Hub.

Migration is a big step to take so it's something that you should plan carefully. We provide guidance and tools to help you with this process.

- While you are migrating your production instance, we advise that you run your current mobile instance and Mobile Hub in parallel for a while, probably a month.
- The migration process should follow the following cycle, migrate > test > migrate, where you perform migration steps and test that the data that will be migrated is correct. If necessary, repeat this step a number of times. Finally you perform the migration by running the migration tool.
- An external storage account is used for data transfer.

This guide uses the following terminology:

- Source instance: Refers to the mobile cloud instance running in MCS or OMCe.
- Target instance: Refers to the Mobile Hub.

This guide does not include detailed procedures on the migration of Oracle WebLogic Server users, groups, or policies from the source instance to the target instance. This includes users and groups that are defined in the default WebLogic Server authentication provider (embedded LDAP), as well as users and groups that exist in an external identity provider like an LDAP server. Refer to the migration (export/import) capabilities of your identity provider.

The processes of migrating from MCS to Mobile Hub and migrating from OMCe to Mobile Hub are slightly different so be sure to read the correct information for your source instance.

You can migrate to Mobile Hub from:

- OMCe versions 18.2.1, 18.2.2, 18.2.3, or 18.2.5
- MCS version 19.1.3

Learn About Oracle Mobile Hub

Mobile Hub is a cloud-based service that provides a unified hub for developing, deploying, maintaining, monitoring, and analyzing your mobile apps and the resources they rely on. As a service administrator, you can use this guide to provision, monitor, and maintain Mobile Hub instances.

Mobile Hub is a suite of public cloud services that consists of a set of integrated components that enables developers, managers, and mobile cloud administrators to develop, maintain, monitor, and export your mobile apps and the resources they rely on.

Find out more about how Mobile Hub provides a unified hub for developing, deploying, maintaining, monitoring, and analyzing your mobile apps and the resources that they rely on. See [Developing Applications with Oracle Mobile Hub](#).

Plan your Migration Strategy

Follow these steps to migrate to Mobile Hub:

- You need a new instance of Mobile Hub ready to use. See [Getting Started](#).
- You also need an object storage instance of 10Gb so that data can be migrated. The migration process uses Object Storage or Storage Classic for the data that is exported from MCS. Once the data has been imported to Mobile Hub, you can clean up the storage instance.
- We provide a migration tool, which is a script which runs in a Docker environment. The migration tool for MCS is not the same as the migration tool for OMCe.
- You should plan the migration process carefully and follow a cycle where you test the migration, possibly a number of times, to be sure that it produces the results you expect, and only then perform the actual migration.
- (MCS only) You migrate your users from MCS, a traditional cloud account, to Mobile Hub, where users are stored in Identity Cloud Service.

This table describes the task flow for migrating to Mobile Hub.

| Task | Description | How Do I Do This? |
|--|---|--|
| Get ready to migrate | Download the migration tool | The Migration Tool |
| (OMCe only) Patch the instance of OMCe | If you are migrating from OMCe, you need to apply a patch to the service | Patch OMCe |
| Configure the migration tool | Update the JSON file with the appropriate values for your source instance, the target instance, and the storage instance you'll use for migration | MCS: Configure config.json for MCS OMCe: Configure config.json for OMCe |
| Build the Docker image | Create a Docker image and create the container based on it | Build the Docker Container |
| Do a test runs of the migration tool | Check that the configuration file is correct by doing test runs that don't update Mobile Hub | Test the Migration |
| Migrate to Mobile Hub | Once you are satisfied, run the migration tool for the final time | Perform the Migration |

| Task | Description | How Do I Do This? |
|--|---|---|
| (MCS only) Migrate users to Mobile Hub | Export users from your traditional cloud accounts used by MCS, modify the resulting CSV file, and import users into IDCS used by Mobile Hub | Migrate Users If the data to be migrated includes user-isolated storage collection objects, users must be migrated before data is migrated |

Data Types

The migration tool exports the following types of data from the source instance (MCS or OMCe) to the storage instance, from where it is imported to the target instance, Mobile Hub:

- Mobile Backends
- Client Management
- APIs
- Connectors
- Collections
- Policies
- Location data
- Tables created by Database Access and Management APIs
- Analytics data (MCS only)
- Security certificates and credentials

Roles

You have to have the following roles to perform the migration:

- Be administrator role on the MCS or OMCe instance, and on the Mobile Hub instance
- For the storage instance, you need to:
 - Be administrator on the tenant that uses the object storage
 - Or, have the policy to use the object storage. For example:

```
Allow user userOCID to manage objects in tenancy
```

```
Allow group MigrationAdmins to manage objects in compartment OMHMigration
```


2

The Migration Tool

The migration tool is supported on Linux and Mac.

The migration tool is different for OMCe and for MCS. Download the appropriate one from the Mobile Downloads page at <https://www.oracle.com/downloads/cloud/mobile-cloud-service-downloads.html>. Save the zip to a local location and unzip it.

The migration tool has four migration files:

- `config.json` This is the file which you edit to specify details of the source instance, the storage account, and the target Mobile Hub.
- Three files which you do not edit:
 - `Dockerfile`
 - `functions.py`
 - `migrate.py`

The information you use in the configuration file is used to:

- Generate export bundles which are exported to the storage instance.
- import the generated bundles from the storage instance to the Mobile Hub instance.

The OMCe migration tool also contains patch scripts that you use to patch the OMCe instance before performing the migration:

- `patch_OMCE-6385_OMCe18.2.1.sh` patch script for OMCe 18.2.1
- `patch_OMCE-6385_OMCe18.2.2.sh` patch script for OMCe 18.2.2
- `patch_OMCE-6530_OMCe18.2.3.sh` patch script for OMCe 18.2.3
- `patch_OMCE-6529_OMCe18.2.5.sh` patch script for OMCe 18.2.5

Patch OMCe

If you are migrating from OMCe to Mobile Hub you need to patch the OMCe instance using a patch script.

If you are migrating from MCS ignore this section and go to [About the Configuration File](#).

Choose the appropriate patch script for the your OMCe instance:

- Edit the script so that it can access the admin server using WLST
- Save the patch script to the Core VM
- Run the patch script

You can find the OMCe version by clicking the user icon, and then select About.

Edit the Patch Script

Edit the patch script to add the Core Instance WLS admin server credentials.

```
##### for APPLY operation
export WLS_ADMIN_USER="<OMCe admin user>"
export WLS_ADMIN_PASSWORD="<password for OMCe admin user>"
export WLS_ADMIN_SERVER_HOST="<WLS admin server host>"
export WLS_ADMIN_SERVER_PORT="7001"
```

Save the patch file with the original name.

The WLS credentials are Administrator Username and Administrator Password you used when you first provisioned OMCe. See *Set Up the OMCe Service in Administering Oracle Mobile Cloud Enterprise in a Customer-Managed Environment*. The default admin username is `omceadmin`.

Alternatively, you can try to retrieve the credentials.

1. Find the Core VM's IP address.
 - a. From OMCe Stack Overview, click the resource with the name `<stack>CORE`.
 - b. In the Service Overview for the Mobile Core POD Details, expand Resources and note the Public IP, for example `192.0.2.1`.
2. Connect to the Core VM using `ssh` and change to the `oracle` user. You can log in as the default user, `opc`. The `opc` user has `sudo` privileges.
 - a. If you don't know the private key you can set a new one. See *Add a New SSH Public Key in Administering Oracle Mobile Cloud Enterprise in a Customer-Managed Environment*.

To make the private key not user-readable, use:

```
chmod 600 <private_key_file>
```

- b. Log in using:

```
$ ssh -i <private_key_file> opc@<Core IP address>
```

For example, `ssh -i my_private_rsa_keyfile opc@192.0.2.1`

- c. Switch to the `oracle` user using:

```
$ sudo su - oracle
```

3. Run the following command:

```
[oracle@spomcebc core-wls-1 temp]$ cat /u01/data/domains/mobile/temp/WLSTLauncher.pod.postcreate.output.json | python -c 'import json,sys;obj = json.load(sys.stdin); print obj["SM_SENSITIVE_DATA"] ["encodedString"]' | base64 -d | python -c 'import json,sys;obj = json.load(sys.stdin); print json.dumps(obj["mobileWLSAdminCred_instance"])' | python -m json.tool
```

Sample result:

```
{
  "name": "omceadmin",
  "password": "Welc0me#",
  "type": "USERPASS"
}
```

Save the Patch Script to the Core VM

Connect to the Core VM using ssh.

1. Find the Core VM's IP address.
 - a. From OMCe Stack Overview, click the resource with the name `<stack>CORE`.
 - b. In the Service Overview for the Mobile Core POD Details, expand Resources and note the Public IP, for example `192.0.2.1`.
2. Connect to the Core VM using ssh and change to the `oracle` user. You can log in as the default user, `opc`. The `opc` user has `sudo` privileges.

- a. Use:

```
$ ssh -i <private_key_file> opc@<Core IP address>
```

For example, `ssh -i my_private_rsa_keyfile opc@192.0.2.1`

- b. Switch to the `oracle` user using:

```
$ sudo su - oracle
```

3. Give `exec` permissions for current user, and then you can run the patch script.

Alternatively, you can upload the script file to Core VM using the `scp` tool and using same `<private_key_file>` and `<Core IP address>` as for ssh.

```
$ scp -i <private_key_file> <path to local script>
  opc@<Core IP address>:<path to upload script>
```

Run the Patch Script

The patch scripts support these operations:

- `FIND` determines whether the patch has been applied or not
- `APPLY` applies the patch
- `ROLLBACK` rollsbacks the applied patch

Note:

The examples use `patch_OMCE-6529_OMCe18.2.5.sh`. Make sure you use the correct patch script name when you run the script.

Running the Patch Script

1. Verify that the patch hasn't already been applied.

```
./patch_OMCE-6529_OMCe18.2.5.sh FIND
```

Expected results:

- **FOUND** the patch has been applied to the vm, executing **APPLY** would result in **noop**
- **NOT FOUND** the patch not found on the vm, executing **ROLLBACK** (if defined) would be **noop**
- **HALF_FOUND** executing both **APPLY** and **ROLLBACK** (if defined) would change the state of the vm
- **ERROR** something has gone wrong. See [Troubleshoot Patch Issues](#).

FIND is default operation that will be started if you run patching script without specifying operation parameter.

2. Apply the patch.

```
./patch_OMCE-6529_OMCe18.2.5.sh APPLY
```

The operation performs these steps:

- **Find operation.** Patch application stops if the patch has been already applied
- **Backup.** Every **APPLY** operation creates a copy of artifacts to be patched in backup folder:

- OMCE 18.2.1 /u01/app/oracle/crs/patch/OMCE-6385/data/backup
- OMCE 18.2.2 /u01/app/oracle/crs/patch/OMCE-6385/data/backup
- OMCE 18.2.3 /u01/app/oracle/crs/patch/OMCE-6530/data/backup
- OMCE 18.2.5 /u01/app/oracle/crs/patch/OMCE-6529/data/backup

If backup folder already exists, it is renamed to `backup_<current_date_time>`, for example `backup_2019.08.02-16.46.55`

Expected results:

- **FOUND** the patch has been already applied earlier
- **APPLIED** the operation has applied the patch
- **ERROR** something has gone wrong. See [Troubleshoot Patch Issues](#).

Rolling Back the Patch Script

Rollback is invoked automatically if patch application fails. You can call it manually if the patch was applied successfully but something went wrong.

By default, rollback restores artifacts from the backup directory created by the last **APPLY** operation.

If you need to rollback the patch, use

```
./patch_OMCE-6529_OMCe18.2.5.sh ROLLBACK
```

If you have to restore from older backup, provide the backup folder name in patch script:

```
##### for ROLLBACK operation
# backup directory to restore patched artifacts from
# absolute path or directory name for auto-created backup (backups
created in e.g. /u01/app/oracle/crs/patch/OMCE-6529/data/)
# defaultvalue will backup from the last created backup
export ROLLBACK_FROM_FOLDER=backup
```

Expected results:

- `NOT_FOUND` the patch has not been applied, nothing to do
- `ROLLED_BACK` the operation has rolled back the patch
- `ERROR` something has gone wrong. See [Troubleshoot Patch Issues](#).

Restart the Core VM

In the Stack Details page of the Stack Manager, locate the stack you want to start and select `Start`.

Troubleshoot Patch Issues

If one of the patch operations has returned an error, review the operation logs for operation detailed messages.

The operation logs are appended to the log file stored in the current patch data directory:

- `OMCe 18.2.1` /u01/app/oracle/crs/patch/OMCE-6385/log.txt
- `OMCe 18.2.2` /u01/app/oracle/crs/patch/OMCE-6385/log.txt
- `OMCe 18.2.3` /u01/app/oracle/crs/patch/OMCE-6530/log.txt
- `OMCe 18.2.5` /u01/app/oracle/crs/patch/OMCE-6529/log.txt

The log file contains everything output by the `apply`, `rollback`, and `find` operations, with a line between the output for each operation.

About the Configuration File

`config.json` is a configuration file that you update with details of:

- **Scope** (Optional). Defines set of data to be migrated. If you are trying to identify problems with a test migration it can be useful to restrict the scope.
- **Source Instance** Details of the MCS instance you are exporting the bundles from.
- **Target Instance** Details of the Mobile Hub instance that you are importing the bundles to.

- **Storage Account** Details of the object storage you use for migration.
- **Security** (Optional) A security key to encrypt migrated security data.

This is a sample config.json for OMCE to Mobile Hub. There are some differences between the migration script for OMCE and for MCS so it's important to download and edit the correct version of the migration tool.

```
{
  "///": "defines set of data to be migrated. migrate: true/false value
used to mark data types to be exported. true is default.",
  "scope": {
    "metadata": {
      "///": "migrate artifacts metadata",
      "migrate": true,
      "///": "exclude clients by name or name+version",
      "exclude": ["<client1_name>",
"<client2_name>(<client2_version>)",
      "///": "migrate only default implementations. Default = false",
      "onlyDefaultImpl": false,
      "///": "limit number of implementations stored per zip entry.
Default = 0 means default limit",
      "implPageSize": 0
    },
    "policies": {
      "///": "migrate policies",
      "migrate": true
    },
    "collections": {
      "///": "migrate collection objects. Requires collection metadata
to be also migrated",
      "migrate": true,
      "///": "do not migrate objects for collections with given names",
      "exclude": ["collectionName1", "collectionName2"],
      "///": "migrate only collection objects updated after specified
date. Supported values `-5s`, `-5m`, `-5h` or date format `yyyy-MM-
dd'T'HH:mm:ss.SSS'Z'`.",
      "fromTime": "2019-09-03T00:00:00.000-07:00",
      "///": "If collection table in target instance already exists,
merge new objects into existing table, skipping duplicates (true) or
report conflict and stop (false).",
      "merge": true
    },
    "database": {
      "///": "migrate Database Management Service data: tables, indexes,
constraints",
      "migrate": false,
      "///": "optional attribute to list tables to be migrated",
      "tables": ["tableName1", "tableName2"],
      "///": "optional attribute to list tables to be excluded from
migration",
      "exclude": ["APEX$TEAM_DEV_FILES"]
    },
    "location": {
      "///": "migrate location data: Assets, Places, Devices",
      "migrate": true
    }
  }
}
```

```

    },
    "security": {
      "///": "migrate security data",
      "migrate": false,
      "///": "include Notification Profiles security data",
      "profiles": false
    },
    "analytics": {
      "///": "migrate analytics data. Disabled by default",
      "migrate": false
    }
  },
  "///": "OMCe instance to export data from",
  "source_instance": {
    "instance_url": "http://foo.oraclecloud.com:7777",
    "token_url": "https://foo.storage.oraclecloud.com/auth/v1/token",
    "client_id": "abcde12345abcde12345abcde12345ab",
    "client_secret": "abcde123-abcd-1234-1245-
abcde12345ab",
    "username": "user1",
    "password": "password"
  },
  "///": "OMH instance to import data to",
  "target_instance": {
    "instance_url": "http://foo.oraclecloud.com:7777",
    "token_url": "https://foo.storage.oraclecloud.com/auth/v1/token",
    "client_id": "abcde12345abcde12345abcde12345ab",
    "client_secret": "abcde123-abcd-1234-1245-abcde12345ab",
    "username": "user1",
    "password": "password"
  },
  "///": "storage account, that is used during the migration",
  "storage_account": {
    "X-Storage-BaseUrl": "https://foo.storage.oraclecloud.com",
    "X-Storage-Account": "myservice",
    "X-Storage-Tenant": "myservice",
    "X-Storage-Container": "Migration",
    "X-Storage-User": "Storage user",
    "X-Storage-Password": "Storage password"
  },
  "///": "additional security configuration"
  "security": {
    "///": "Optional user-defined password to enable security data
encryption",
    "X-Data-EncryptionKey": "UserPassword"
  }
}

```

Configure config.json for MCS

Before running the migration tool, you have to update the `config.json` with the appropriate information for your source MCS instance, the target Mobile Hub, and the storage instance used by the migration process.

Migration Scope

By default, migration is enabled for all data types except analytics data. You can define the set of data to be migrated by choosing to enable or disable each data type using `true` or `false`, as shown in this example. This can be useful when you are trying to isolate an issue that is causing a test migration to fail.

```
{
  "//": "defines set of data to be migrated. migrate: true/false value
used to mark data types to
    be exported. true is default.",
  "scope": {
    "metadata": {
      "//": "migrate artifacts metadata",
      "migrate": true
    },
    "policies": {
      "//": "migrate policies",
      "migrate": true
    },
    "collections": {
      "//": "migrate collection objects. Requires collection metadata
to be also migrated",
      "migrate": true,
      "//": "Optional attribute. Do not migrate
objects for collections with given names",
      "exclude": ["collectionName1", "collectionName2"]
    },
    "database": {
      "//": "migrate Database Management Service data: tables, indexes,
constraints",
      "migrate": false,
      "//": "optional attribute to list tables to be migrated",
      "tables": ["tableName1", "tableName2"],
      "//": "optional attribute to list tables to be excluded from
migration",
      "exclude": ["APEX$TEAM_DEV_FILES"]
    },
    "location": {
      "//": "migrate location data: Assets, Places, Devices",
      "migrate": true
    },
    "security": {
      "//": "migrate security data",
      "migrate": false,
      "//":
        "include Notification Profiles security data",
      "profiles": false
    },
    "analytics": {
      "//": "migrate analytics data. Disabled by default",
      "migrate": false
    }
  },
}
```


Source Instance

The section of `config.json` where you enter details of the MCS instance.

```
"/": "MCS instance to export data from",
  "source_instance": {
    "instance_url": "<YOUR_BASE_URL>",
    "token_url": "<YOUR_OAUTH_TOKEN_ENDPOINT>",
    "client_id": "<YOUR_CLIENT_ID>",
    "client_secret": "<YOUR_CLIENT_SECRET>",
    "username": "<YOUR_MCS_USERNAME>",
    "password": "<YOUR_MCS_PASSWORD>",
    "identity_domain": "<YOUR_IDENTITY_DOMAIN>"
  }
```

1. In MCS, open **Applications > Backends**, then open any Mobile Backend.
2. Select **Settings**.
3. Under **Environment URLs**, use the Base URL value for `instance_url` and the OAuth Token Endpoint value for `token_url`.
4. Under **Access Keys** under **OAuth Consumer**, use the Client ID value for `client_id` and the Client Secret value for `client_secret`.

Target Instance

The section of `config.json` where you enter details of the Mobile Hub instance.

```
"/": "OMH instance to import data to",
  "target_instance": {
    "instance_url": "<YOUR_BASE_URL>",
    "token_url": "<YOUR_OAUTH_TOKEN_ENDPOINT>",
    "client_id": "<TEAM_MEMBER_APP_CLIENT_ID>",
    "client_secret": "<TEAM_MEMBER_APP_CLIENT_SECRET>",
    "username": "<YOUR_MOBILE_HUB_USERNAME>",
    "password": "<YOUR_MOBILE_HUB_PASSWORD>"
  }
```

1. To find the Team Member App Client ID and Team Member App Client Secret, in Mobile Hub open **Development** and click on **Instance Details**.
2. To find the `instance_url` and `token_url`:
 - a. In Mobile Hub, open **Development > Backend**.
 - b. Select **Settings**.
 - c. Under **Environment URLs** find the Base URL value which you use for `instance_url`.
Find the OAuth Token Endpoint value which you use for `token_url`.

Storage Account

The section of `config.json` where you enter details of the Object Storage or Storage Classic instance you are using.

```
"//": "storage account, that is used during the migration",
  "storage_account": {
    "X-Storage-BaseUrl": "<OBJECT_STORAGE_URL>",
    "X-Storage-Account": "<STORAGE_ACCOUNT>",
    "X-Storage-Tenant": "<TENANT_NAME>",
    "X-Storage-Container": "<CONTAINER_NAME>",
    "X-Storage-User": "<STORAGE_USERNAME>",
    "X-Storage-Password": "<STORAGE_PASSWORD>"
  }
}
```

Using Object Storage



Note:

The Storage Swift password is referenced in Oracle Cloud Infrastructure (OCI) as an access token. See [Working with Auth Tokens](#). If the Auth Token contains < you must generate a new token. Auth Tokens containing < are rejected by the security layer.

1. Login to OCI Console.
2. In the upper right choose the **region** you use for Object Storage, for example, `us-ashburn-1`.
3. The value for `X-Storage-BaseUrl` is the hostname of the Swift API for the region you have chosen. Go to [API Reference and Endpoints](#) and expand Object Storage and Archive Storage then Swift API. Use the hostname for the region you have used, e.g. `https://swiftobjectstorage.us-ashburn-1.oraclecloud.com`.
4. Open the User menu and choose **Tenancy**. From the Tenancy Information tab, set `X-Storage-Tenant` to be **Name**.
5. Set `X-Storage-Account` to be **Object Storage Namespace**.
6. From the Menu (upper left) choose **Object Storage > Object Storage**. Either use an existing bucket, or create a new bucket. Set `X-Storage-Container` to be **Name** for the bucket.

If login to OCI Object Storage using the Swift API fails with 401 then you may need to add policies so that the user accessing the bucket is part of the group which has a policy defined to access that bucket or all objects in the tenancy.

Using Storage Classic

1. Go to **My Services** and then to **Oracle Cloud Infrastructure Object Storage Classic**.
Go to **Account** and open the **Account Information** page.

2. In **Account Information** find the **Rest Endpoint**. This has the format `<BaseUrl>/v1/<account>`.
The `X-Storage-BaseUrl` is the `<BaseUrl>`.
`X-Account` and `X-Storage-Tenant` are both the same and are `<account>`.
3. `X-Storage-Container` is the **Container** value. The container will be created if it does not exist.

Security

Optionally use a password to encrypt security data during migration.

```

"//": "additional security configuration"
"security": {
  "//": "Optional user-defined password to enable security data
encryption",
  "X-Data-EncryptionKey": "UserPassword",
}

```

Configure config.json for OMCe

Before running the migration tool, you have to update the JSON configuration file with the appropriate information for your source OMCe instance, the target Mobile Hub, and the storage instance used by the migration process.

Migration Scope

By default, migration is enabled for all data types except analytics data. You can define the set of data to be migrated by choosing to enable or disable each data type using `true` or `false`, as shown in this example. This can be useful when you are trying to isolate an issue that is causing a test migration to fail.

```

{
  "//": "defines set of data to be migrated. migrate: true/false value
used to mark data types to
be exported. true is default.",
  "scope": {
    "metadata": {
      "//": "migrate artifacts metadata",
      "migrate": true
    },
    "policies": {
      "//": "migrate policies",
      "migrate": true
    },
    "collections": {
      "//": "migrate collection objects. Requires collection metadata
to be also migrated",
      "migrate": true,
      "//": "Optional attribute. Do not migrate
objects for collections with given names",
      "exclude": ["collectionName1", "collectionName2"]
    }
  }
}

```

```

    },
    "database": {
      "///": "migrate Database Management Service data: tables, indexes,
constraints",
      "migrate": false,
      "///": "optional attribute to list tables to be migrated",
      "tables": ["tableName1", "tableName2"],
      "///": "optional attribute to list tables to be excluded from
migration",
      "exclude": ["APEX$TEAM_DEV_FILES"]
    },
    "location": {
      "///": "migrate location data: Assets, Places, Devices",
      "migrate": true
    },
    "security": {
      "///": "migrate security data",
      "migrate": false,      "///":
      "include Notification Profiles security data",
      "profiles": false
    }
  },
},

```

Source Instance

The section of config.json where you enter details of the OMCE instance.

```

"///": "OMCE instance to export data from",
"source_instance": {
  "instance_url": "<YOUR_BASE_URL>",
  "token_url": "<YOUR_OAUTH_TOKEN_ENDPOINT>",
  "client_id": "<TEAM_MEMBER_APP_CLIENT_ID>",
  "client_secret": "<TEAM_MEMBER_APP_CLIENT_SECRET>",
  "username": "<YOUR_OMCE_USERNAME>",
  "password": "<YOUR_OMCE_PASSWORD>"
}

```

1. To find the Team Member App Client ID and Team Member App Client Secret, in OMCE open **Development** and click on **Instance Details**.
2. To find the instance_url and token_url:
 - a. In Mobile Hub, open **Development > Backend**.
 - b. Select **Settings**.
 - c. Under **Environment URLs** find the Base URL value which you use for instance_url.
Find the OAuth Token Endpoint value which you use for token_url.

Target Instance

The section of `config.json` where you enter details of the Mobile Hub instance.

```
"/": "OMH instance to import data to",
  "target_instance": {
    "instance_url": "<YOUR_BASE_URL>",
    "token_url": "<YOUR_OAUTH_TOKEN_ENDPOINT>",
    "client_id": "<TEAM_MEMBER_APP_CLIENT_ID>",
    "client_secret": "<TEAM_MEMBER_APP_CLIENT_SECRET>",
    "username": "<YOUR_MOBILE_HUB_USERNAME>",
    "password": "<YOUR_MOBILE_HUB_PASSWORD>"
  }
```

1. To find the Team Member App Client ID and Team Member App Client Secret, in Mobile Hub open **Development** and click on **Instance Details**.
2. To find the `instance_url` and `token_url`:
 - a. In Mobile Hub, open **Development** > **Backend**.
 - b. Select **Settings**.
 - c. Under **Environment URLs** find the Base URL value which you use for `instance_url`.

Find the OAuth Token Endpoint value which you use for `token_url`.

Storage Account

The section of `config.json` where you enter details of the Object Storage or Storage Classic instance you are using.

```
"/": "storage account, that is used during the migration",
  "storage_account": {
    "X-Storage-BaseUrl": "<OBJECT_STORAGE_URL>",
    "X-Storage-Account": "<STORAGE_ACCOUNT>",
    "X-Storage-Tenant": "<TENANT_NAME>",
    "X-Storage-Container": "<CONTAINER_NAME>",
    "X-Storage-User": "<STORAGE_USERNAME>",
    "X-Storage-Password": "<STORAGE_PASSWORD>"
  }
```

Using Object Storage

Note:

The Storage Swift password is referenced in Oracle Cloud Infrastructure (OCI) as an access token. See [Working with Auth Tokens](#). If the Auth Token contains < you must generate a new token. Auth Tokens containing < are rejected by the security layer.

1. Login to Oracle Cloud Infrastructure (OCI) Console.
2. In the upper right choose the **region** you use for Object Storage, for example, us-ashburn-1.
3. The value for X-Storage-BaseUrl is the hostname of the Swift API for the region you have chosen. Go to [API Reference and Endpoints](#) and expand Object Storage and Archive Storage then Swift API. Use the hostname for the region you have used, e.g. https://swiftobjectstorage.us-ashburn-1.oraclecloud.com.
4. Open the User menu and choose **Tenancy**. From the Tenancy Information tab, set X-Storage-Tenant to be **Name**.
5. Set X-Storage-Account to be **Object Storage Namespace**.
6. From the Menu (upper left) choose **Object Storage > Object Storage**. Either use an existing bucket, or create a new bucket. Set X-Storage-Container to be **Name** for the bucket.

Using Storage Classic

1. Go to Infrastructure Classic Console and then to **Oracle Cloud Infrastructure Object Storage Classic**.
Go to **Account** and open the **Account Information** page.
2. In **Account Information** find the **Rest Endpoint**. This has the format <BaseUrl>/v1/<account>.
The X-Storage-BaseUrl is the <BaseUrl>.
X-Account and X-Storage-Tenant are both the same and are <account>.
3. X-Storage-Container is the **Container** value. The container will be created if it does not exist.

Security

Optionally use a password to encrypt security data during migration.

```
  "///": "additional security configuration"
  "security": {
    "///": "Optional user-defined password to enable security data
encryption",
    "X-Data-EncryptionKey": "UserPassword",
  }
```

3

Migrate to Oracle Mobile Hub

Oracle recommends that you simulate running the migration tool from end to end, but without modifying the OMH target. That is, creating the export bundles but not importing them into the target Mobile Hub. This allows you to check that the values in the `config.json` file are correct, and that there are no conflicts in the data.

The migration tool runs in Docker, an open platform for building, shipping, and running distributed applications. It gives programmers, development teams, and operations engineers the common toolbox they need to take advantage of the distributed and networked nature of modern applications. You'll need to have Docker installed on your system to run the migration tool.

For more information about Docker, see the [Docker documentation](http://www.docker.com) at www.docker.com.

Build the Docker Container

After updating the `config.json` for your source instance, you build the Docker container so that you can run the migration tool.

Use the **docker build** command to create a Docker image. It uses the definition contained in the `Dockerfile` which is part of the migration tool.

```
docker build -t mcs-migration .
```

Alternatively, if you are using a proxy:

```
docker build \  
  --build-arg "http_proxy=http://www-proxy.example.com:80" \  
  --build-arg "https_proxy=http://www-proxy.example.com:80" \  
  -t mcs-migration .
```

In both examples, there is a dot at the end which is a shortcut for the current directory, assuming that you are working in the directory where the migration tool was unzipped.

The Docker image is ready when you see `Successfully built`.

Test the Migration

Every time you run `migrate` a new migration bundle is created and uploaded (exported) into storage. You can use commands that specify a bundle to identify issues when an export (to storage) or an installation or import (from storage to the target instance) fails.

1. Run the test migration using:

```
docker run -it mcs-migration python migrate.py migrate -n
```

2. Check the output.

- If the overall status is `SUCCESS`, you can proceed to run the migration.
- If the status shows `FAILED` or `PART_FAILURE`, examine the output to see where the failure occurred.

3. • If export failed, show the status of the export using:

```
docker run -it mcs-migration python migrate.py export_status
bundle_name
```


- If install failed, show the status of the installation using:

```
docker run -it mcs-migration python migrate.py install_status
bundle_name
```

Resolve any conflicts that are reported.

Migration Tool Commands

| Command | Description |
|--|---|
| <code>docker run -it mcs-migration python migrate.py migrate -n</code> | Run to export to a storage bundle without modifying the target Mobile Hub instance. |
| <code>docker run -it mcs-migration python migrate.py export_status <bundle_name></code> | Run to show the export status after <code>FAILED</code> or <code>PART_FAILURE</code> . |
| <code>docker run -it mcs-migration python migrate.py install_status <bundle_name></code> | Run to show the installation or import status after <code>FAILED</code> or <code>PART_FAILURE</code> . |
| <code>docker run -it mcs-migration python migrate.py migrate</code> | Execute migration from the source instance to Mobile Hub. |
| <code>docker run -it mcs-migration python migrate.py -h</code> | Run to see help. |
| <code>docker run -it mcs-migration python migrate.py export</code> | Export data from the source instance to storage. |
| <code>docker run -it mcs-migration python migrate.py list</code> | List the bundles available in storage for installation in the Mobile Hub instance. |
| <code>docker run -it mcs-migration python migrate.py install <bundle_name> -n</code> | Simulate installing or importing the specified bundle from storage to Mobile Hub during a test migration. |
| <code>docker run -it mcs-migration python migrate.py install <bundle_name></code> | Install or import the specified bundle from storage in the target Mobile Hub instance. |

| Command | Description |
|---|--|
| <code>docker run -it mcs-migration python migrate.py migrate -n -s</code> | <p>(Migration from MCS only.)</p> <p>When testing the migration, excludes the following from migration:</p> <ul style="list-style-type: none"> • Metadata for clients having same set of platform, version and Bundle ID (other names Application ID or Package Name) in the source instance. These are not supported in Mobile Hub and conflicts might not be resolved in the MCS production environment. • User-Isolated Collections tables that cannot be migrated because of missing IDCS users in the Mobile Hub instance. You should migrate users from the traditional cloud account (MCS) to Oracle Identity Cloud Service (for Mobile Hub) before performing the migration. |
| <code>docker run -it mcs-migration python migrate.py migrate -s -d</code> | <p>(Migration from MCS only.)</p> <p>As above, but dismisses objects that have no owner in IDCS.</p> |
| <code>docker run -it mcs-migration python migrate.py migrate -s</code> | <p>(Migration from MCS only.)</p> <p>As above, but when running the end-to-end migration.</p> |
| <code>docker run -it mcs-migration python migrate.py cleanup</code> | <p>(Migration from MCS only.)</p> <p>Use to cleanup the target instance in case of conflicts.</p> <p>This command removes all supported artifacts, including demo data. Oracle recommends you only use this for testing migration.</p> |
| | <div style="border-left: 2px solid #d4ac3d; padding-left: 10px; margin-bottom: 10px;"> <p> WARNING:</p> <p>Only run this on an Mobile Hub instance that has nothing you care about on it, since everything will be deleted.</p> </div> <p>The supported types which are removed are:</p> <ul style="list-style-type: none"> • metadata • location data • collection objects for migrated collections |
| <code>docker run -it mcs-migration python migrate.py cleanup_storage</code> | <p>Every migrate operation uploads a new migration bundle into configured Object Storage and leaves it there so you can examine it for conflicts or other issues. Use this task to remove all bundles.</p> |

| Command | Description |
|---|---|
| <pre>docker run -it mcs-migration python migrate.py cleanup_storage <bundle_name></pre> | Use to clean up (remove) a specific bundle. |

Perform the Migration

When you have tested that the migration tool is working correctly, you can migrate data from the source instance to Mobile Hub.

Run this command:

- `docker run -it mcs-migration python migrate.py migrate`

Bundles are created in the source instance and exported to the storage instance. Then they are imported to the target Mobile Hub.

The migration is complete when the overall status is `SUCCESS`.

In Mobile Hub, check that you can see the correct:

- Mobile Backends
- Client Management
- APIs
- Connectors
- Collections
- Policies
- Location data
- Tables created by Database Access and Management APIs
- Analytics data (migration from MCS only)

Use a Custom config.json

You can run a migration tool command using a custom `config.json` without rebuilding the docker image using this command and appropriate arguments:

```
docker run -v ${PWD}/config.json:/migration/config.json -it  
mcs-migration python migrate.py <arguments>
```

User-Isolated Collection Objects (MCS Only)

When the data to be migrated includes user-isolated storage collection objects, users need to be migrated from the traditional cloud account (MCS) to IDCS (for OMH) before the migration tool is run. Otherwise, migration will fail because the associated users are not present in the OMH instance.

4

Security Configuration (MCS Only)

The way security is handled in MCS is very different compared to how it is done in Mobile Hub. Mobile Hub uses Oracle Identity Cloud Service (IDCS) as the identity and access management infrastructure and it provides Mobile Hub applications with:

- Identity and access management
- Tokens

Depending on the complexity of your existing Oracle Mobile Cloud configurations there may be additional tasks you have to perform so that your mobile applications continue to work once you have migrated to Mobile Hub.

At a minimum, you should migrate:

- Users
- Application role memberships
- Admin users

Migrate Users

To migrate users, first, export them from the traditional cloud accounts. Then, modify the CSV file that contains the users you exported so that you can import them into Oracle Identity Cloud Service. Next, import the users into Oracle Identity Cloud Service.

When you're migrating users to cloud accounts with Identity Cloud Service, you may want users to use the passwords from their traditional cloud accounts. To do this, change the minimum length of the custom password policy in Oracle Identity Cloud Service to eight characters.

Export users from Traditional Cloud Accounts

1. Sign in to the Infrastructure Classic Console of the traditional cloud account that contains the users that you want to export.
2. Click the **Users** tab. The Identity Cloud Service console appears.
3. Click **Users**.
4. Select the users that you want to export.
5. Click **Export**. The users will be exported into a CSV file.
6. In the dialog box that appears, save the CSV file to your machine.

Edit the Users File

1. Open the CSV file in an appropriate editor and make the following changes:
 - Rename the `User Login` column to `User ID`.

- Rename the `Email` column to `Work Email`.
2. Save the updated CSV file.

Import Users into Oracle Identity Cloud Service

1. Sign in to the Infrastructure Classic Console of the cloud account with Identity Cloud Service.
2. Click the **Users** tab. The Identity Cloud Service console appears.
3. Expand the **Navigation Drawer**, and then click **Users**.
4. Click **Import**.
5. In the **Import Users** dialog box, click **Browse** to locate and select your CSV file.
6. Verify that the path and name of the CSV file you selected appear in the **Select a file to import** field.
7. Click **Import**.
8. After Oracle Identity Cloud Service evaluates all users, review the job results.
 - If the job can be processed immediately, then a dialog box appears with the **Job ID** link for your import job. Click the link and review the details that appear on the **Jobs** page.
 - If the job can't be processed immediately, then a message appears with a Schedule ID in it. Copy that Schedule ID, and use it to search for the job on the **Jobs** page. The job will appear when processing completes. Go to step 9.
9. In the **Jobs** page, locate the job that you want to view, and then click **View Details**.

A table displays the first names, last names, email addresses, user names, and statuses of the users that you imported into Oracle Identity Cloud Service.
10. Review the details that appear on the **Jobs** page.

This page shows how many users you imported, how many users you imported successfully, and how many users can't be imported because of a system error.

Each user that you successfully imported receives a Welcome email notification. The user can use the link in this notification to access their account and set a password for it.