

Cloudera AI 1.5.4

Cloudera AI Project Migration

Date published: 2022-12-14

Date modified: 2025-03-18

CLOUDERA

<https://docs.cloudera.com/>

Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

Migrating Projects.....	4
Troubleshooting: Using custom SSL certificates to connect to Cloudera AI Workbenches.....	7
Troubleshooting: Retrying a successful or failed migration.....	8
Troubleshooting: Running the tool as a background process.....	8
Project Migration FAQs.....	8

Migrating Projects

You can migrate individual projects from CDSW 1.10.x to Cloudera AI both on cloud and on premises environments.

The Cloudera AI command-line utility seamlessly migrates projects across different environments, such as from CDSW to Cloudera AI. The migration includes project files stored on NFS, project settings, models, jobs, and applications. The utility currently supports migration from CDSW 1.10 to Cloudera AI public cloud version 2.0.40 onwards, as well as from CDSW 1.10 to Cloudera AI on premises versions 2.0.39 (1.5.1-CHF1) onwards.

Project migration requires a third machine (user laptop or bastion host) with connectivity to both CDSW and Cloudera AI. The project is first downloaded from CDSW to this intermediate machine using the `export` command, and then it is uploaded to the Cloudera AI Workbench using the `import` command. The utility uses the `cdswctl` client for login and creation of an ssh session and tunnel. It uses `rsync` for migration of project files through this tunnel. Project artifacts (models/jobs/applications) are migrated using APIs.

Authentication is carried out using the API key provided during migration, and only authorized users are allowed to migrate projects. The data in transit will remain encrypted as long as the workbench has an `https` connection.

Prerequisites

A third machine (user laptop or bastion host) shall have the following configuration:

- Unix-like system (macOS or Linux).
- Connectivity to both CDSW and Cloudera AI.
- `Rsync` should be installed.
- Python version ≥ 3.10 .
- Sufficient disk space to hold project contents. For stronger security, the disk and/or file system should be encrypted.
- Set up custom CA certificates if required (Check FAQ section).

Migration prerequisites:

- All custom runtimes shall be configured on the target workbench prior to running project migration.
- All the users/collaborators/teams should be migrated to the target workbench, and appropriate access should be provided to these users.
- The user migrating the project should be an Admin/Owner/Collaborator of the project. Note that the user migrating the project will become owner of the project in the target Cloudera AI Workbench.
- Migration of projects created in a team context is supported, provided the team is already created in the target workbench. Each team member has the right to migrate such projects.
- Please note that settings or configurations outside the project are not migrated. These include:
 - User quota and custom quota
 - Security configurations
 - SMTP settings
 - Public ssh keys
 - Ephemeral storage settings
 - Kerberos configuration
 - Environment variables
 - Engine/Runtime configuration

The steps enumerated below apply to both the source and target workbenches:

1. Make sure that you have an `rsync`-enabled runtime image (`cloudera/ml-runtime-workbench-python3.9-standard-rsync`) added to your Cloudera AI Workbench runtime catalog. The image is hosted on [DockerHub](#). If the `rsync` image is not readily accessible, it can be created from a Dockerfile hosted [here](#) and hosted in any registry. If not, ask your workbench admin to prepare the Dockerfile.

2. Make sure that your default `ssh` public keys are added under your user settings. If not, please add them in `User Settings Remote Editing SSH public keys` for session access. The default `ssh` public key shall be available on your machine at `~/.ssh/<something>.pub`. The SSH public key must be uploaded to both Cloudera AI or CDSW source and target workbenches.
3. If an `ssh` key pair does not already exist in your system, create one.
4. It is recommended to avoid setting a passphrase for the `ssh` key, because there are multiple instances in which an `ssh` connection is established. If a passphrase is set, automation using the utility would be tedious.
5. The Legacy API key must be available and noted down. You will need this API key during migration. To generate a new key, head over to `User Settings API Keys Legacy API key`.

Legacy engine migration

Projects using the legacy engine can be migrated to engine-based projects by moving the legacy engine to ML Runtime mapping in the `cmlutils/constants.py` file. The `LEGACY_ENGINE_MAP` in the `constants.py` file shall be an empty map for this. The workloads that uses the legacy engine or custom engine images will be migrated to the default engine image in the destination cluster.

Steps for project migration

1. Install the utility on a third machine or bastion host: `python3 -m pip install git+https://github.com/cloudera/cmlutils@main`
2. To export the project, create the `export-config.ini` file inside the `<home-dir>/cmlutils` directory.
3. Inside the `export-config.ini` file, create a section for each project, where you can include project-specific configurations. For common configurations shared across projects, place them in the `DEFAULT` section.

```
[DEFAULT]
url=<Source-Workspace-url>
output_dir=~/Documents/temp_dir
ca_path=~/Documents/custom-ca-source.pem

[Project-A]
username=user-1
apiv1_key=umxma76ilel6pgm36zacrx2bywakflvz

[Project-B]
username=user-2
apiv1_key=fmxma76ilel6pgm36zacrx2bywaklopq
```

Configuration used:

- `username`: username of the user who is migrating the project. (Mandatory)
 - `url`: Source workbench URL (Mandatory)
 - `apiv1_key`: Source API v1/Legacy API key (Mandatory)
 - `output_dir`: temporary directory on the local machine where the project data/metadata would be stored. (Mandatory)
 - `ca_path`: path to a CA (Certifying Authority) bundle to use, in case python is not able to pick up CA from the system and `ssl` certificate verification fails. Issue is generally seen with MacOS. (Optional)
4. If you wish to skip certain files or directories during export, create `.exportignore` file at the root of the CDSW project (`/home/cdsw`). The `.exportignore` file follows the same semantics as that of `.gitignore`.

5. Run the following project export command

```
cmlutil project export -p "Project-A"
```

or

```
cmlutil project export -p "Project-B"
```



Note: The project name shall match one of the section names in the export-config.ini file.

6. A folder with the project name is created inside the output directory (~/Documents/temp_dir). If the project folder already exists, then the data is overwritten.
7. All the project files, artifacts and logs corresponding to the project are downloaded in the project folder.
8. Create the import-config.ini file inside the <home-dir>/cmlutils directory.
9. Inside the import-config.ini file, create a section for each project, where you can include project-specific configurations. Place common configurations shared across projects in the DEFAULT section.

Example file:

```
[DEFAULT]
url=<Destination-Workspace-url>
output_dir=~/Documents/temp_dir
ca_path=~/Documents/custom-ca-target.pem

[Project-A]
username=user-1
apiv1_key=abcma76ilel6pgm36zacrx2bywakflvz

[Project-B]
username=user-2
apiv1_key=xyzma76ilel6pgm36zacrx2bywaklopq
```

Configuration used:

- username: username of the user who is migrating the project. (Mandatory)
- url: Target workbench URL (Mandatory)
- apiv1_key: Target API v1/Legacy API key (Mandatory)
- output_dir: temporary directory on the local machine from where the project will be uploaded. (Mandatory)
- ca_path: path to a CA (Certifying Authority) bundle to use, in case python is not able to pick up CA from the system and ssl certificate verification fails. Issue is generally seen with macOS. (Optional)

10. Run the following project import command

```
cmlutil project import -p "Project-A"
```

or

```
cmlutil project import -p "Project-B"
```



Note: The project name shall match one of the section names in the import-config.ini file.

11. The project is created in the destination workbench, if it does not exist already. Projects with the same name are overwritten.

Post migration guidelines

- In the target workbench, the user's public SSH key will be different from the source. Remember to update the SSH key in all external locations, such as the github repository.

- After the migration, the Model API key and endpoint URL are different from the source. Ensure that you update all applications that utilize these APIs.
- All the Models/Jobs/applications are created in paused or stopped state in the destination workbench, so all the artifacts shall be restarted post migration. Before starting the Models, Jobs or Application in the destination workbench, the corresponding workloads shall be stopped in the source workbench to avoid any data corruption if both are accessing the same data.
- Any configuration parameters outside the project configuration shall be copied manually after the migration.

Batch migration

- The Cloudera AI Utility is primarily designed to facilitate the migration of individual projects. However, there is a wrapper script available that enables batch migration of multiple projects. Two Python scripts are available, one for export and another for import.
- The batch migration script reads the list of project names from the export-config.ini and import-config.ini files. Each section defined here corresponds to a specific project, with the section name corresponding to the project name. You can include project-specific configurations within each respective section, while configurations shared across multiple projects can be placed inside the "default" section.
- The BatchSize variable provided inside the script controls the number of projects that can be exported or imported simultaneously. To prevent system errors like running out of memory, it is essential to select an appropriate batch size. Each export or import operation of a project generates a distinct session on the workbench, utilizing 1 CPU and 0.5 GB of memory. Therefore, the batch size shall be determined considering the available resources on both the source and target workbenches.
- Before initiating the batch migration, ensure that enough disk space is available on the host machine for downloading all or a batch of projects.
- In case of failure during batch migration, the script can be rerun. However, to speed up the execution of the batch it is recommended to delete all the project-names already exported or imported from the configuration file.
- Logs for each project are collected inside the individual project directory.

Troubleshooting: Using custom SSL certificates to connect to Cloudera AI Workbenches

The migration script internally uses `cdswctl` which relies on system-wide configured certificates to establish connection with source and target workbenches. It must be ensured that SSL certificates belonging to both source and target workbenches (if applicable) shall be set properly.

Set up for Linux (Ubuntu/Debian)

1. Copy your CA to the directory `/usr/local/share/ca-certificates/`. Ensure that the CA file is in `.crt` format.
2. Run the command: `sudo cp foo.crt /usr/local/share/ca-certificates/foo.crt`
3. Update the CA store: `sudo update-ca-certificates`.

Set up for MacOS

1. Navigate to `Finder Applications Utilities Keychain Access`.
2. Select `System` in the left-hand column.
3. Open `File Import Items` and import the certificate files into the system keychain.
4. The certificate shall now appear with a red 'X'. That means it is entrusted. To provide trust, double-click the certificate. Under `Trust`, change the setting at the top when using this certificate to `Always Trust`.



Note: For the utility to run with custom certificates on macOS, add the certificate to the keychain and pass the certificate (`ca_path`) inside the `export-config.ini` and `import-config.ini` files. This is because Python is not able to pick up system-wide certificates in macOS.

Troubleshooting: Retrying a successful or failed migration

The utility has been carefully designed to resume the export or import operation from the exact point where it left off in the event of a failure.

Project files

The utility employs rsync to facilitate the migration of project files. When the export/import command is rerun, it will synchronize the project files from the source to the destination.

Project settings and artifacts

During a rerun, the project settings and artifacts (model/job/application) that have already been migrated are not updated. Instead, only the missing artifacts in the target workspace are migrated. The utility is not designed to support a sync operation between the source and target projects. If you wish to update the project that has already been migrated, it is advisable to delete the project in the target workspace and then rerun the migration.

Troubleshooting: Running the tool as a background process

There might be cases where the tool is required to run as a separate process detached from the terminal. In that case please use this command on your Linux or macOS machine:

```
nohup <complete CLI command> > <stdout-file> 2> <stderr-file> &
```

This prints out the Process ID (PID) and appends the logs to the specified stdout and stderr file locations.

The process can be interrupted by sending a SIGINT to the PID returned. Please use `kill -INT <PID>` in that case.

Project Migration FAQs

Some frequently asked questions about using the project migration tool.

Q: I do not want to migrate all project data into new Cloudera AI Workbenches. How do I ignore/skip certain files and directories?

A: The CLI tool recognises a file named `.exportignore` in which you can specify the files/directories either in full, or through patterns. You have to create this file at the root of the CDSW or Cloudera AI project (`/home/cdsw`) before running the export sub-command. The `.exportignore` file follows the same semantics as that of `.gitignore`.

Q: How to fix the error "REMOTE HOST IDENTIFICATION HAS CHANGED!" or "HOST KEY VERIFICATION FAILED"?

A: Remove the host key entry from the file `~/.ssh/known_hosts` using command `ssh-keygen -R server-hostname-or-IP`. For example: `ssh-keygen -R '[localhost]:5104'`

Q: How to fix the error "SSL: CERTIFICATE_VERIFY_FAILED"?

A: To resolve this error, create a bundle of all root or intermediate Certification Authorities (CAs) in `.pem` format. Save this bundle file and provide its path (`ca_path`) inside the `export-config.ini` and `import-config.ini` files. This will ensure that Python can locate the trusted certificate and prevent the error from occurring.

Q: How to fix the error "configparser.NoSectionError: No section: 'sample_project'" ?

A: It means the utility cannot find a section for the "sample_project" inside the `export-config.ini` or `import-config.ini` files.