Keycloak in Docker #5 – How to export a realm with users and secrets

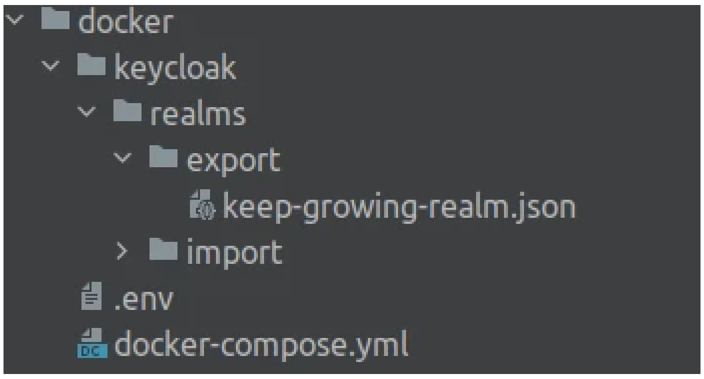
2nd February 2022 / little_pinecone / Tools

Running a Keycloak service in a Docker container allows us to share its configuration across multiple environments. However, we can also export an entire Keycloak realm in case we need any backups or data transfer between servers.

Prerequisites

- Docker Engine and Docker Compose installed on your machine.
- If this is your first attempt to run Keycloak in Docker, I recommend reading the post Keycloak in Docker # 1 – How to run Keycloak in a Docker container as I explained the basic configuration there.

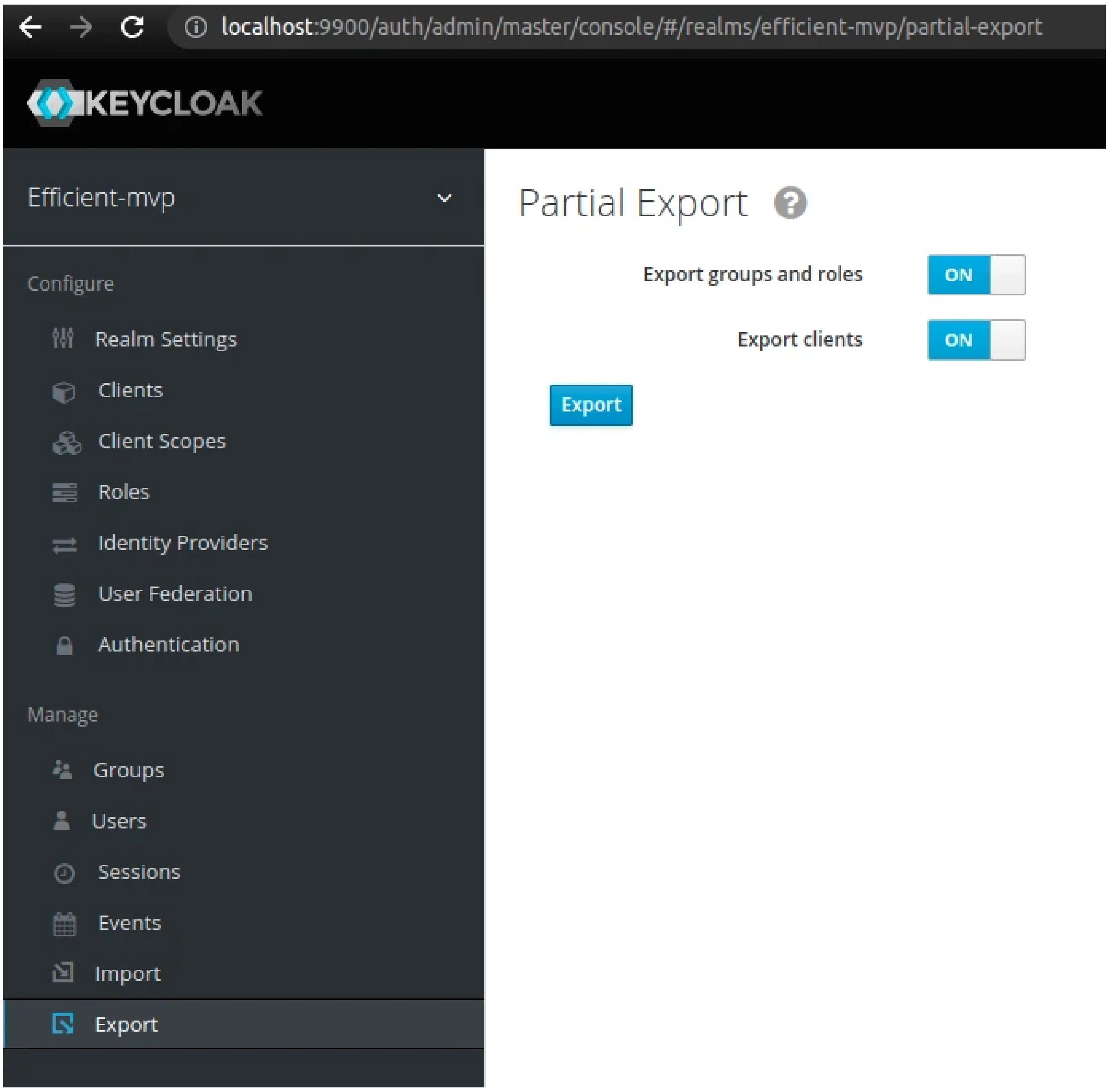
For reference, below you'll find the resulting directory tree for this example:



```
docker
  keycloak
    realms
      export
        keep-growing-realm.json
      import
    .env
    docker-compose.yml
```

In short, my goal is to get the **`export`** directory containing realm resources (in this case, the **`keep-growing-realm.json`** file). Thanks to this, I'll be able to backup or recreate a complete realm whenever I need to.

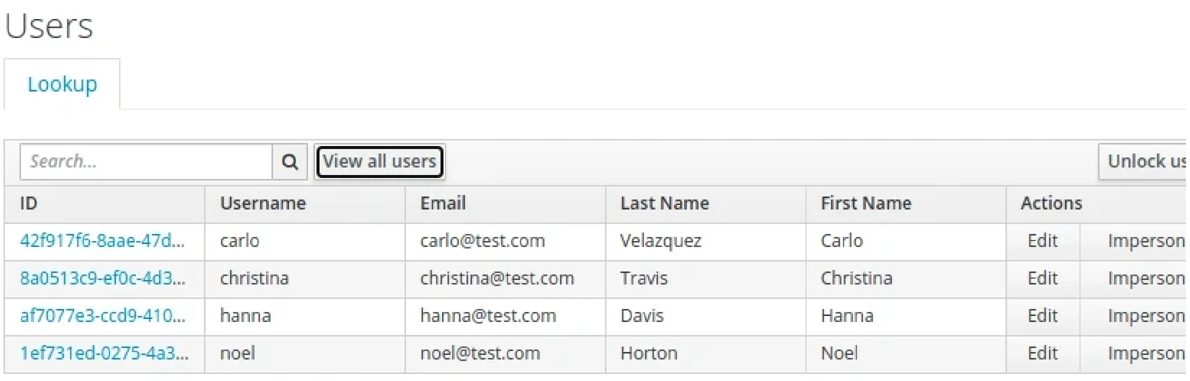
The Keycloak Admin Console provides an easy way to export a realm. However, as you can see in the screenshot below, not all resources can be exported with this method:



Export groups and roles: ON
Export clients: ON

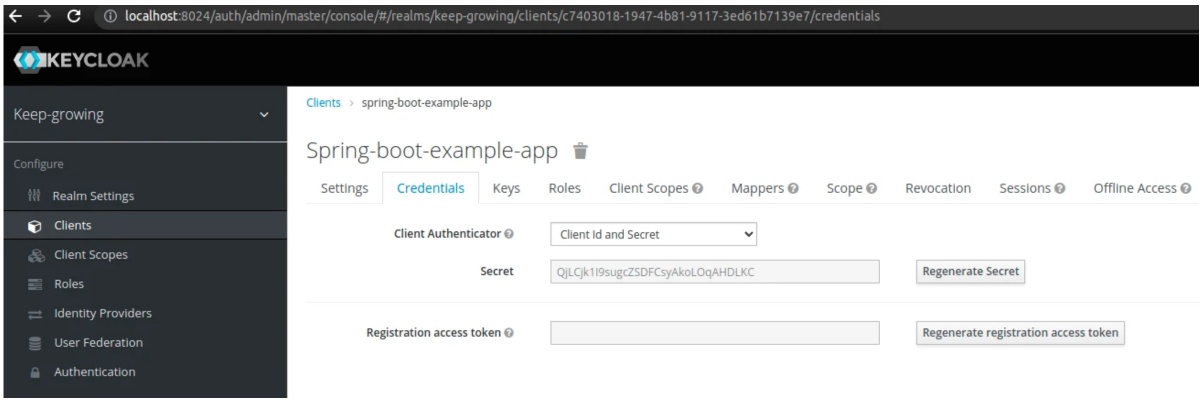
The `realm-export.json` file produced with this technique won't contain user data. In addition, client secrets will be masked. Although this approach might be appropriate in some use cases, we won't be able to recreate the same instance using only this file.

If we want to get realm data suitable for backups or cross server migration, we need to run a boot-time export.



ID	Username	Email	Last Name	First Name	Actions	
42f917f6-8aae-47d...	carlo	carlo@test.com	Velazquez	Carlo	Edit	Imperson
8a0513c9-ef0c-4d3...	christina	christina@test.com	Travis	Christina	Edit	Imperson
af7077e3-ccd9-410...	hanna	hanna@test.com	Davis	Hanna	Edit	Imperson
1ef731ed-0275-4a3...	noel	noel@test.com	Horton	Noel	Edit	Imperson

Furthermore, the realm contains the `spring-boot-example-app` client. It has the confidential access type and a secret, as you can see in the screenshot below:



Client Authenticator: Client Id and Secret
Secret: QjLCjk1I9sugcZSDFCsyAkoLOqAHDLKC
Registration access token:

At the end of this article, I will verify that the users and the secret are properly exported.

How to export all resources from a Keycloak realm

I'm going to use a volume to map exported data between the container and my machine. Next, I will show you how to export the data with a command or automatically when running the container. Finally, I'm going to verify if the export was successful.

Add a volume for exported resources

```yaml
version: '3.3'
services:
  keycloak:
    image: jboss/keycloak:${KEYCLOAK_VERSION}
    ports:
      - "8024:8080"
    environment:
      - KEYCLOAK_USER=${KEYCLOAK_USER}
      - KEYCLOAK_PASSWORD=${KEYCLOAK_PASSWORD}
    volumes:
      - ./keycloak/realms/export:/tmp/export
```

I'm using my default environmental variables defined in the following `.env` file:

```
KEYCLOAK_VERSION=16.1.1
KEYCLOAK_USER=keycloak
KEYCLOAK_PASSWORD=keycloak
```

Now I'm going to start the service with the `docker-compose up -d` command. As we can see in the following screenshot, the volume is properly created and mapped:

Obviously, the `/tmp/export` directory inside the dockerized Keycloak instance is empty as we still need to export our realm:

Keycloak export options explained

Let me introduce the properties that we're going to use:

- `Djboss.socket.binding.port-offset`, we want to run the export on a different port than Keycloak itself. By default, the jboss server in my `kecyloak` instance uses the `9990` port. Therefore, providing e.g. 100 as a value here will result in the command using the `10090` port.
- `Dkeycloak.migration.action`, we specify what we want to do. Available options:
  - `export`,
  - `import`.
- `Dkeycloak.migration.provider`, we define how we want the data. Available options are:
  - `singleFile`, if we want to export data into a file
  - `dir`, if we want to export data into a directory.
- `Dkeycloak.migration.realmName`, realm for export. Don't specify this parameter if you want to export all realms.
- `Dkeycloak.migration.usersExportStrategy`, how to export realm users. Available options are:
  - DIFFERENT_FILES, it's the default option for this attribute, we will get a file for realm config and multiple files for users (depending on how many users we have and the number of users per one file (50 by default));
  - SKIP, won't export users;
  - REALM_FILE, export users to the same file as realm configuration;

mapping.

Run the export command

We're going to execute the `/opt/jboss/keycloak/bin/standalone.sh` script included in the jboss/keycloak image to initiate the export process:

```
docker exec -it <CONTAINER NAME> /opt/jboss/keycloak/bin/standalone.sh \
… // required Keycloak export properties
```

To clarify the following examples, my example `keycloak` service runs in the `keycloakspringboot_keycloak_1` container.

Export to a single file

When a realm doesn't contain a lot of users it might be appropriate to export it into a single file. The actual command that I'm going to execute looks like in the snippet below:

```
docker exec -it keycloakspringboot_keycloak_1 /opt/jboss/keycloak/bin/standalone
-Djboss.socket.binding.port-offset=100 \
-Dkeycloak.migration.action=export \
-Dkeycloak.migration.provider=singleFile \
-Dkeycloak.migration.realmName=keep-growing \
-Dkeycloak.migration.usersExportStrategy=REALM_FILE \
-Dkeycloak.migration.file=/tmp/export/keep-growing-realm.json
```

If the command encounters no problems, you'll see console output similar to this:

```
…
12:11:10,399 INFO  [org.keycloak.services] (ServerService Thread Pool -- 60) K
C-SERVICES0034: Export of realm 'keep-growing' requested.
12:11:10,400 INFO  [org.keycloak.exportimport.singlefile.SingleFileExportProvi
der] (ServerService Thread Pool -- 60) Exporting realm 'keep-growing' into fil
e /tmp/export/keep-growing-realm.json

```

```
12:11:11,196 INFO  [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin
console listening on http://127.0.0.1:10090
```

As a result, a new file appears in the **`keycloak/realms/export`** directory on my machine:

You can quit the process with **`Ctr+C`**.

If you want to import that realm, the Keycloak in Docker #2 – How to import a Keycloak realm article describes in detail importing a realm from a file.

Export to a directory

Keycloak recommends exporting data into a directory if your realm contains more than 500 users:

> Exporting many users into a directory performs optimally as the directory provider uses a separate transaction for each "page" (a file of users).
>
> *https://www.keycloak.org/docs/16.1/server_admin/#assembly-exporting-importing_server_administration_guide*

In this case, my command would look like this (notice changes in lines 4 and 6):

```
1 docker exec -it keycloakspringboot_keycloak_1 /opt/jboss/keycloak/bin/standalone
2 -Djboss.socket.binding.port-offset=100 \
3 -Dkeycloak.migration.action=export \
4 -Dkeycloak.migration.provider=dir \
5 -Dkeycloak.migration.realmName=keep-growing \
6 -Dkeycloak.migration.dir=/tmp/export
```

If the command encounters no problems, you'll see console output similar to this:

```
12:29:32,212 INFO  [org.keycloak.exportimport.dir.DirExportProvider] (ServerSe
rvice Thread Pool -- 53) Users 0-3 exported
12:29:32,214 INFO  [org.keycloak.services] (ServerService Thread Pool -- 53) K
C-SERVICES0035: Export finished successfully
…
```

As a result, a group of new files appear in the `keycloak/realms/export` directory on my machine:

You can quit the process with `Ctr+C`.

If you want to import that realm, see the Keycloak in Docker #6 – How to import realms from a directory article as it describes in detail importing realms exported to a folder.

Export on a container startup

Additionally, we can configure our `docker-compose.yml` file to initiate the export when we start the container (by adding the command config). It can be useful when our docker image doesn't provide an equivalent to the `standalone.sh` script or we just don't want to use it. The modified compose file for a single file export looks like this:

```yaml
version: '3.3'
services:
  keycloak:
    image: jboss/keycloak:${KEYCLOAK_VERSION}
    ports:
      - "8024:8080"
    environment:
      - "-Dkeycloak.migration.action=export"
      - "-Dkeycloak.migration.provider=singleFile"
      - "-Dkeycloak.migration.realmName=keep-growing"
      - "-Dkeycloak.migration.usersExportStrategy=REALM_FILE"
      - "-Dkeycloak.migration.file=/tmp/export/keep-growing-realm.json"
```

We'll see in logs that the container starts with the provided options:

The export will proceed as in the previously described methods.

Data included in the exported Keycloak realm resources

We're going to find the secret generated for the `spring-boot-example-app` client in the realm config file:

Furthermore, the users are included in the exported assets as well. In case of the directory export the users are stored in the `keep-growing-users-0.json` file. For the single file export, the users are included directly in the `keep-growing-realm.json` file:

Troubleshooting

What to check when the export didn't work as planned?

Null pointer exception

If you see the `NullPointerException` error in the logs make sure that the `Dkeycloak.migration.realmName` property value is consistent with the actual name of the realm you want to export. The following example error shows what happens if I provide `non-existing` to the command but my realm's id is actually `keep-growing`:

```
11:42:20,795 INFO  [org.keycloak.services] (ServerService Thread Pool -- 54) KC-SERVICES0034: Export of realm 'non-existing' requested.
11:42:20,795 INFO  [org.keycloak.exportimport.singlefile.SingleFileExportProvider] (ServerService Thread Pool -- 54) Exporting realm 'non-existing' into file /tmp/export/keep-growing-realm.json
11:42:20,804 FATAL [org.keycloak.services] (ServerService Thread Pool -- 54) Error during startup: java.lang.NullPointerException
```

No such file or directory

If you see the `No such file or directory` message in the logs make sure that the `Dkeycloak.migration.file` property value points to a path that actually exists within the container. The following example error shows what happens if I provide `/tmp/wrong/directory/path/keep-growing-realm.json` to the command but Docker actually created a folder according to the mapped volume which points to `/tmp/export`:

Verify that you actually restarted the container after adding the volume config to the `docker-compose.yml` file. Otherwise, the mapped directory won't be created for you.

Missing export files

There are no errors in logs but you can't find the exported files on your machine? Verify that the `Dkeycloak.migration.file` property value is consistent with the volume mapping. The following example shows what happens if I provide `/tmp/keep-growing-realm-test.json` instead of `/tmp/export/keep-growing-realm-test.json`:

You can always verify the location of the exported data in the command logs:

```
11:59:32,192 INFO  [org.keycloak.services] (ServerService Thread Pool -- 53) KC-SERVICES0034: Export of realm 'keep-growing' requested.
11:59:32,192 INFO  [org.keycloak.exportimport.singlefile.SingleFileExportProvider] (ServerService Thread Pool -- 53) Exporting realm 'keep-growing' into file /tmp/keep-growing-realm-test.json
11:59:32,805 INFO  [org.keycloak.services] (ServerService Thread Pool -- 53) KC-SERVICES0035: Export finished successfully
```

- The Jboss image documentation on exporting a realm
- The Keycloak documentation on importing and exporting the database

Photo by RODNAE Productions from Pexels

Docker Keycloak user management

2 thoughts on "Keycloak in Docker #5 – How to export a realm with users and secrets"

Syed says:

5th April 2022 at 2:44 pm

Hi,
While running command to export in folder, getting below error
/tmp/export/testing-realm-realm.json (Permission denied)

```
3:41:45,821 INFO  [org.hibernate.orm.beans] (ServerService Thread Pool — 59)
HHH10005002: No explicit CDI BeanManager reference was passed to Hibernate,
but CDI is available on the Hibernate ClassLoader.
13:41:46,076 INFO  [org.hibernate.validator.internal.util.Version] (ServerService
Thread Pool — 59) HV000001: Hibernate Validator 6.0.22.Final
13:41:47,433 INFO  [org.hibernate.hql.internal.QueryTranslatorFactoryInitiator]
(ServerService Thread Pool — 59) HHH000397: Using ASTQueryTranslatorFactory
13:41:48,485 INFO  [org.keycloak.services] (ServerService Thread Pool — 59) KC-
SERVICES0034: Export of realm 'testing-realm' requested.
13:41:49,063 FATAL [org.keycloak.services] (ServerService Thread Pool — 59) Error
during startup: java.lang.RuntimeException: Error during export/import:
/tmp/export/testing-realm-realm.json (Permission denied)
at org.keycloak.keycloak-
services@17.0.1//org.keycloak.exportimport.util.ExportImportSessionTask.run(Expo
rtImportSessionTask.java:37)
at org.keycloak.keycloak-server-spi-
private@17.0.1//org.keycloak.models.utils.KeycloakModelUtils.runJobInTransaction
(KeycloakModelUtils.java:239)
at org.keycloak.keycloak-
services@17.0.1//org.keycloak.exportimport.util.MultipleStepsExportProvider.expor
```

drwxr-xr-x 2 jboss root 4096 Apr 5 13:41 hsperfdata_jboss
drwxr-xr-x 1 root root 4096 Mar 30 13:42 hsperfdata_root
-rwx—— 1 root root 291 Mar 28 09:18 ks-script-c9hd6mir
-rwx—— 1 root root 701 Mar 28 09:18 ks-script-jrnjxyf4

Reply

little_pinecone says:

7th April 2022 at 9:14 pm

The permission problem seems to be due to the fact that the directory `export` belongs to the user `root`. In my configuration, the owner is `jboss`:

```
drwxrwxr-x 2 jboss 1000 4096 Feb 3 12:52 export
drwxr-xr-x 2 jboss root 4096 Apr 7 20:04 hsperfdata_jboss
drwxr-xr-x 1 root root 4096 Jan 25 15:26 hsperfdata_root
```

The similar exception will be thrown when the volume on your local machine is owned by the `root`.

Leave a Reply

Your email address will not be published. Required fields are marked *

☐ By using this form you agree with the storage and handling of your data by this website. *

POST COMMENT

Created by Marta Szymek

Privacy Policy