Exporting users from Keycloak (GitHub Edition)

The `v1` of https://github.com/THE-Engineering/export-from-keycloak-import-to-auth0 performed a machine-to-machine export from Keycloak but it was written for local Docker (and I expected to necessarily change that part)

I wrote `v2` on Thursday, disposing of the machine-to-machine export, and instead having it fetch *encrypted* JSON files from a GitHub repository

The application doesn't care how the JSON gets into GitHub — provided it has the location and credentials it will just pull the contents and operate on it

But!

I also made sure that *the pieces* exist for how you might get that JSON into GitHub

This document describes a process for exporting from Keycloak and putting encrypted JSON files into GitHub

Obviously it describes some *manual steps* which you may prefer to automate. I have written all of this for a human but you may prefer to remove them from the process

You will need

- Shell access to the container running Keycloak
- The same `CRYPTO_KEY` that will be used by https://github.com/THE-Engineering/export-from-keycloak-import-to-auth0 when it is deployed
- A GitHub repository configured for Git LFS

It will help to have a **working knowledge of Git LFS** https://git-lfs.com/

Step 1

You are going to generate a directory of JSON files in Keycloak and download them

I put the scripts for exporting from Keycloak into their own repository

https://github.com/THE-Engineering/export-from-keycloak

You can use any of the three shell scripts — they achieve the same thing slightly differently

- `different-files.sh` exports realms and users to different files
- `realm-file.sh` groups realms and users by realm
- `same-file.sh` puts all realms and users into the same file

Either put your script of choice onto the container to execute it, or paste it at the container's command line

I have no idea how long the export will take, but given that UMS alone has more than a million users *I would suggest not writing more than a million files to the file system using* `different-files.sh`

I think `same-file.sh` is a safe enough choice (as is `realm-file.sh`, too, which produces a handful of files, rather than just one)

Each of these shell scripts creates a log as it executes so you can manually read that to verify what's happening — check the script for its location

- Each of the scripts has a timeout of 24 hours (after which it will kill the process). You can modify that to your preference
- Each script writes data to a default location on the container but you can change that, too
- Of course, you will need write permissions to a location somewhere on the container
- Ensure to `chmod +x` the script on the container if necessary (unless of course you paste it at the command line)

As indicated in the repository, the script of your choice will print a message to indicate it has started and that it has finished

If in doubt, as mentioned above, read the log file produced by the script to verify what's happening

Once the export is complete

- Delete the shell script from the container's file system
- Download the directory of JSON file or files on your local device
- Delete them from the container, too

Step 2

You are going to encrypt the directory of JSON files you have just downloaded from Keycloak and push them into GitHub

You will need the same `CRYPTO_KEY` value that will be used by https://github.com/THE-Engineering/export-from-keycloak-import-to-auth0 when it is deployed

Clone https://github.com/sequencemedia/crypto to a location on your file system

Change into that directory. There are two shell scripts of interest

- `encrypt.sh`
- `decrypt.sh`

You will need

- The `CRYPTO_KEY`
- The file path of the directory of JSON files you have just downloaded
- The file path of another directory for the encrypted versions

Replacing the values here, at the command line encrypt the files

```
CRYPTO_KEY='CHANGE ME' ./encrypt.sh \
  --origin "./directory-of-downloaded-JSON-files-CHANGE-ME" \
  --destination "./directory-of-encrypted-JSON-files-CHANGE-ME"
```

Commit the encrypted versions of these files and push them into GitHub

For reference, you can decrypt the files at the command line, too

```
CRYPTO_KEY='CHANGE ME' ./decrypt.sh \
  --origin "./directory-of-encrypted-JSON-files-CHANGE-ME" \
  --destination "./directory-of-decrypted-JSON-files-CHANGE-ME"
```

Step 3

Deploy https://github.com/THE-Engineering/export-from-keycloak-import-to-auth0 into production

It will need

- The `CRYPTO_KEY`
- The location of the GitHub repository containing the encrypted files
- A personal access token to read from the GitHub repository

It will pull the files from GitHub, decrypt them, then operate on them