Authentication

Table of Contents:

Overview

OpenCGA Catalog is an **authenticated environment**. This means that a typical user would need to be authenticated in order to start working with it.

Prior to the deployment, the administrator will set the user registration policy (open to the public or restricted) in the main configuration.yml file (see Configuration section). If the administrator defines a public user registration, any user with access to the webservices will be able to create their own user and start working. However, if restricted, only the administrator will be able to register new user accounts. This can be easily modified by changing the configuration file and restarting the service.

User creation

As for version 1.x, the administrator has two different ways of defining the account type of a user (FULL or GUEST). Basically, a user with a full member account will be able to define its own projects and studies, whereas a user with a guest member account will not. However, it is worth mentioning that despite users with guest accounts are not able to define new projects, they will still be able to create new files, samples... as long as they have the proper creation permissions in other else's user study. The permissions are explained in Catalog Permissions.

There exists basically two different ways of defining new user accounts depending on the authentication system the user will be authenticated against.

Catalog

OpenCGA Catalog have a built-in authentication mechanism implemented. By default, all users will be registered and authenticated against this authentication system. Passwords are safely stored encrypted in our database.

If the user registration is closed to the public, administrators will need to run their own command line to create a user as shown below:

User registration example

```
opencga-admin.sh users create --name John --u john --email john@mail.com --
user-password
```

Import users from other authentication system

Apart from the default mechanism, OpenCGA supports LDAP authentication system. Users and groups that exist in an external LDAP environment can be used in OpenCGA. In this case, if the LDAP mechanism has been properly configured by the OpenCGA administrator, OpenCGA Catalog will be able to authenticate any LDAP user automatically (even if those users were not registered in Catalog), although they will not be allowed to see any data unless they have been granted the proper permissions.

However, although every LDAP user will be able to login, there are a couple of issues that might arise from this behaviour that would need to be understood. The very first time an LDAP user is logged in, OpenCGA will first register that user in the database because OpenCGA really needs to keep track of all the users that might have access to the data.

Example: Let's say that 'owner' is the owner of one study and 'owner' wants to give 'read' permissions to one user that has never logged in OpenCGA before. There are then two ways to grant that user access:

- First, the user logs in (registering the user in the database automatically). Second, 'owner' grants permission to the user because user already exists in OpenCGA.
- The administrator imports the user from an external authentication origin (LDAP) first. Second, 'owner' grants permission to the user because user already exists in OpenCGA.

Import users from LDAP

```
opencga-admin.sh users import --auth-origin {originId} --user {userId}
```

Administrators can also import groups of users running that same command line with different parameters. They could also even create a group and associate those users to that group in a study if it does not exist already:

Import groups of users

```
# Import groups of users
opencga-admin.sh users import --auth-origin {originId} --group
{ldapGroupName}

# Import a group (ldapGroupName) of users, create a group
(newCatalogGroup) within a study (study) containing the users imported
opencga-admin.sh users import --auth-origin {originId} --group
{ldapGroupName} --study-group {newCatalogGroup} --study {study}
```

On the other hand, in an attempt to make things more automatic and easy for users, administrators can make OpenCGA to automatically synchronise groups of users from OpenCGA with groups of users from LDAP for a concrete study. In other words, let's say that we have one "analysts" group defined in LDAP. We will probably want the users belonging to that group to always have the same permissions in a concrete study and, obviously, if we add or remove users from that LDAP group, we will want those users to be granted or revoked permissions accordingly. This can be easily achieved with the previously described feature. To do so, administrators will need to run a different command line where they will need to define one LDAP group and the OpenCGA Catalog group it will be syncronised with:

Group synchronisation

```
opencga-admin.sh users sync --auth-origin {originId} --from
{ldapGroupName} --to {catalogGroupName} --study {study}
```

From that moment on, OpenCGA will do a couple of checks every time an LDAP user logs in:

- Recover from LDAP the groups the user belongs to at that time.
- Remove or add that user from/to all the corresponding synchronised Catalog groups.

In summary, if a user has been removed from an LDAP group that is synchronised with a Catalog group, that user will be removed from the Catalog the moment the user logs. And vice versa, if a user has been added to an LDAP group that is synchronised with any Catalog group, the user will be automatically added to the Catalog group the moment the user logs in.

Login system

As mentioned in the overview, Catalog is an authenticated environment. Because users can only exist in Catalog itself but they could also come from other external authentication systems, we need to store each user's authentication method. This is stored in the user data model, in the account object. Catalog users will be authenticated using our implementation as we have the encrypted password stored. However, users coming from an external authentication system will be authenticated against it.

OpenCGA uses JSON Web Tokens (JWT) standard to provide users with a unique and digitally signed token. The default expiration time of the tokens is 60 minutes but can it can be easily modified in the configuration file. When the authentication is successful, a JWT token is generated and provided. This token will be needed by the user to perform any action from that moment on.