Using an external authentication origin

Table of Contents:

Configuration

In order to be able to authenticate using other authentication origin credentials, it will be necessary defining some parameter present in the *configuration.yml* file. In this section, it will be possible defining as many authentication origins as needed.

```
authenticationOrigins:
- id: ldap1
  type: LDAP   # At the moment, only LDAP type is supported
  host: ldap://localhost:9000
  options:
    usersSearch: dc=ge,dc=co,dc=uk
    groupsSearch: ou=general,ou=groups,dc=ge,dc=co,dc=uk
- id: ldap2
  type: LDAP  # At the moment, only LDAP type is supported
  host: ldap://localhost:8000
  options:
    usersSearch: dc=ge,dc=co,dc=uk
    groupsSearch: ou=general,ou=groups,dc=ge,dc=co,dc=uk
```

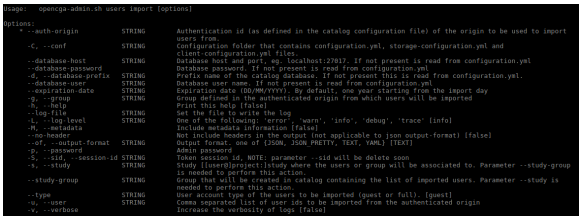
In the below example, we would be defining two different LDAP authentication origins (defined in the type variable). The first one receives the id *ldap1* and the host is in *ldap://localhost:9000,* whereas the second one has the id *ldap2* with a different host.

The *usersSearch* and *groupsSearch* fields are of real importance. In this string fields admins will have to define the naming context to search for users and groups respectively in that authentication origin.

Supported Operations

Once OpenCGA is installed with the proper configuration file, the next step would be adding users from these authenticated origins. To do this, two *admin* command lines have been added to *opencga-admin.sh* script.

Import users



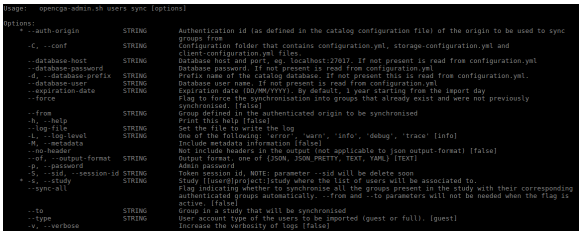
The command line needs the authentication origin id, which in this case would *ldap1* or *ldap2,* and accepts several optional parameters. Admins might opt to provide a list of comma separated users using *-u, --user* and/or a group already defined in their authentication origin containing a list of users that will be directly imported into OpenCGA.

Admins can also define additionally how this new OpenCGA user account will be, the expiration date (*--expiration-date*) or the type (*--type*). There are basically two different types of accounts: guest and full. The main difference between full and guest is that users with a *full* account are able to create their own projects and studies. However, *guest* accounts cannot create anything in OpenCGA unless they have been granted permissions to manipulate **other** user's projects and studies.

Example: Let's imagine that one PI has created a different project/study for every different research the PI is doing. The PI will be able to import other users from an external authentication origin, but most probably, the type of the account the PI will assign to every new user will be *guest*. This way, the PI will be sure other users cannot create anything in OpenCGA. However, the PI will give permissions subsequently to those users. Maybe some users will be able to create new things inside the study, others will only be able to read some information...

The last thing worth explaining is the parameters *--study-group* and *-s, --study*. These parameters will allow the admin to create one group in one study of OpenCGA containing the list of users imported all in one command line.

Sync groups



The aim of this command line is the synchronization of users from one (or more) of the groups from the external authentication origin to one (or more) of the groups defined in one study of OpenCGA.

Basically, this command line can perform two different actions depending on what is already stored in OpenCGA:

Sync and keep track of one group

This method will fetch all the users corresponding to one group defined in the external authentication origin, import the users not previously registered into OpenCGA, create one group in one study (if it did not exist already) and assign those users to the group in OpenCGA (removing other users not belonging to the group in the authentication origin if the group already existed). Besides, additional information will be stored in the group defined in OpenCGA after running this command line, that will let OpenCGA know that that group is *synced* with one particular group from one external authentication origin.

To do so, the mandatory parameters will be:

- *--auth-origin* to define the authentication origin id used to *sync* groups from.
- *-s, --study* to define the study in OpenCGA that has the group to be *synced.*
- *--from* to specify the group from the authentication origin to fetch the current list of users from.
- *--to* to specify the group in OpenCGA whose users have to be *synced.*

Accounts for new user imports can also be defined using the parameters *--expiration-date* and *--type* as explained in the above section.

The command line will complain by default if the admin is trying to *sync* with one existing group in OpenCGA. This can be easily overrided using the parameter *--force.*

Sync all '*synced*' groups from one study

New users might have been added to one group in the external authentication origin and some others might have been removed. However, although it will be implemented, OpenCGA does not resynchronise users from the external authentication origin with the internal group yet. For this reason, it is needed a command line that will check for all the groups that have been imported from external authentication origins and will *resync* the users based on the current members of the external groups.

To do so, the mandatory parameters will be:

- *--auth-origin* to define the authentication origin id used to *sync* groups from.
- *-s, --study* to define the study in OpenCGA that has the group to be *synced.*
- *--sync-all* to indicate that all the groups belonging to the study that have been imported from any group of the authentication origin have to be *resynced.*

This action might require new users to be imported as well. For this reason, the account parameters explained in the sections above might be still necessary.