Table of Contents

What is esgf-getCredentials?

A tool to retrieve user credentials from ESGF. It have one graphic interface and another command line interface.

Getting started

Pre-requisites

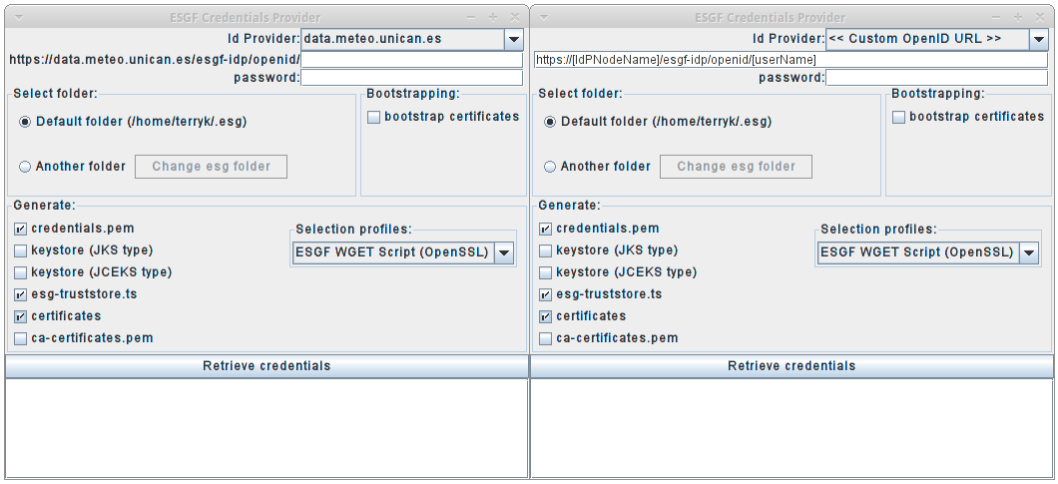
JDK or OpenJDK 6 and upper versions

Download

Download the jar -> getESGFCredentials-0.1.jar 476.4 KB new

Other versions..

Run it



Go to download folder:

- In Windows:
  - Open ESGFToolsUI-v0.8.jar
- Command-line interpreter:

```
java -jar ESGFToolsUI-v0.8.jar
```

Command line UI Guide

Command line help

```
$ java -jar getESGFCredentials-0.1.jar --help
```

Basic usage

```
esgf-getcredentials --openid <openid> [other options]
```

Summary of **options**

}}}

To view specific use cases -->

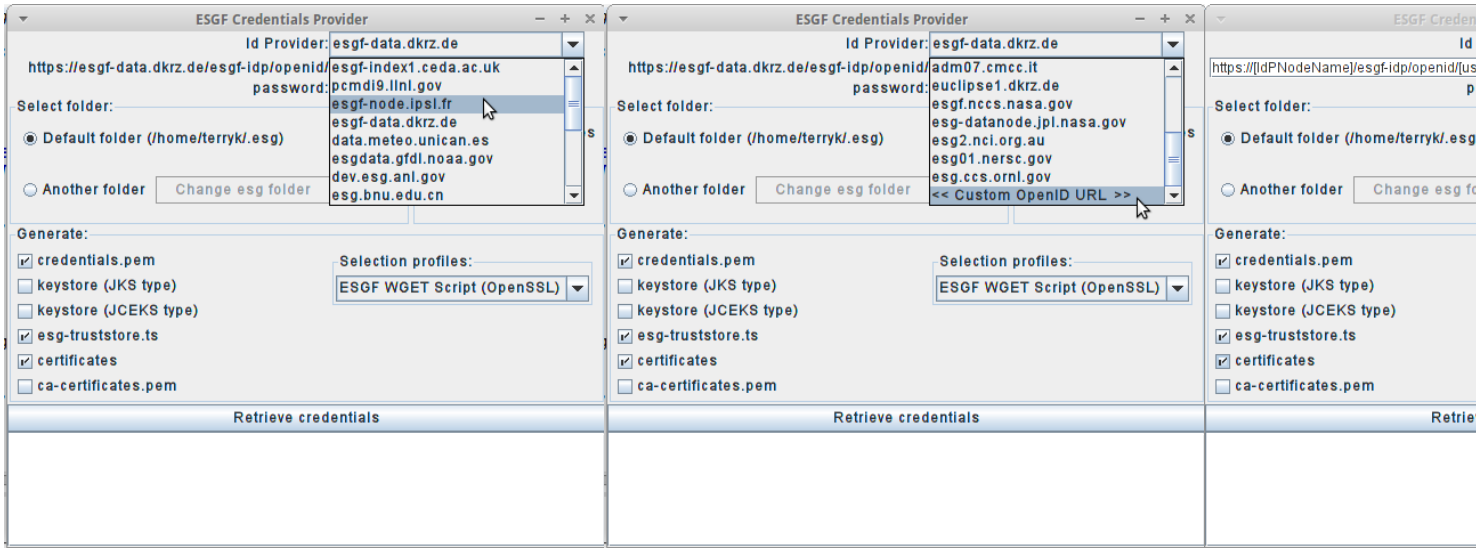
Graphic UI Guide

- In Windows:
  - Open ESGFToolsUI-v0.8.jar
- Command-line interpreter:

```
java -jar ESGFToolsUI-v0.8.jar
```

Setting user

You can select your IdP provider in the top drop-down list. If your IdP provider isn't in the list of providers. Select "Custom OpenID URL", with this option the GUI interface change to be able write OpenID URL's



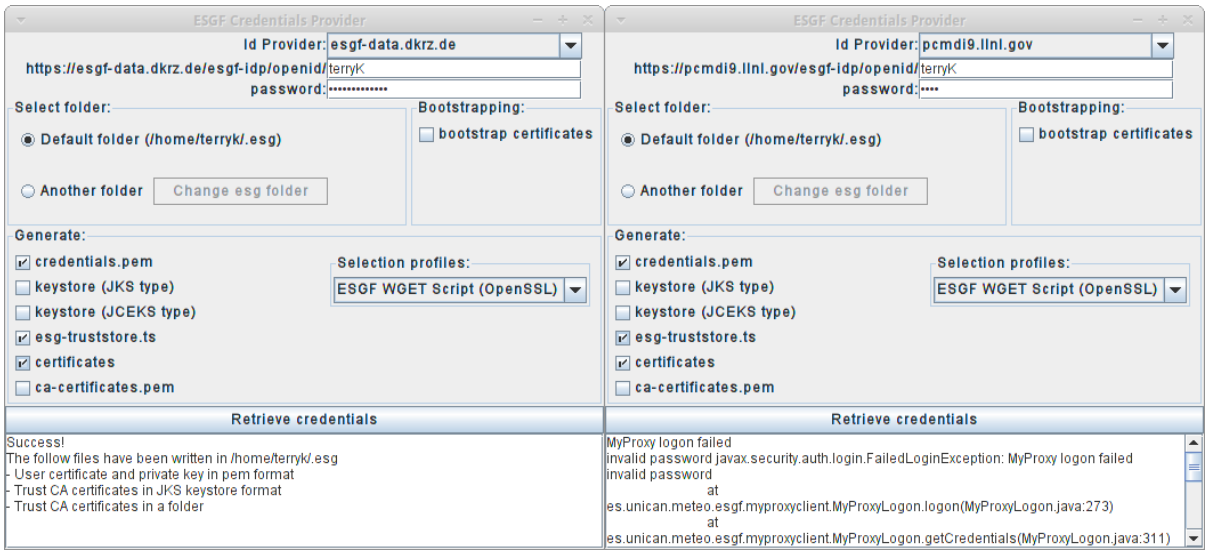
Setting output files

You can select in "Generate" section what output files will be generated in the output folder.

`credentials.pem`	It's a pem file that contains the x509 user certificate and the RSA private key
`keystore (JKS type)`	It's a keystore in format JKS which is build with user cert, cert chain and private key
`keystore (JCEKS type)`	It's a keystore in format JCEKS which is build with user cert, cert chain and private key
`esgf-truststore.ts`	CA's certificates in keystore in format JKS
`certificates`	CA's certificate files and policy files in a folder
`ca-certificates.pem`	CA's certificates in pem format

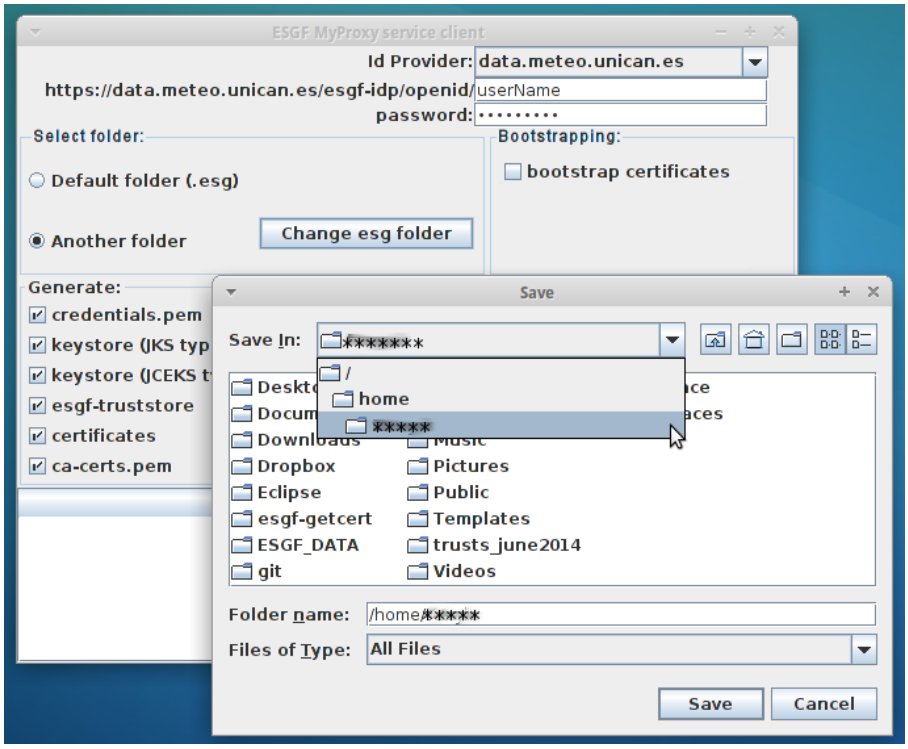
Retrieve credentials

Click on "retrieve credentials" button. If all goes well a success message is shown. However, if some error happens then the Exception is showed



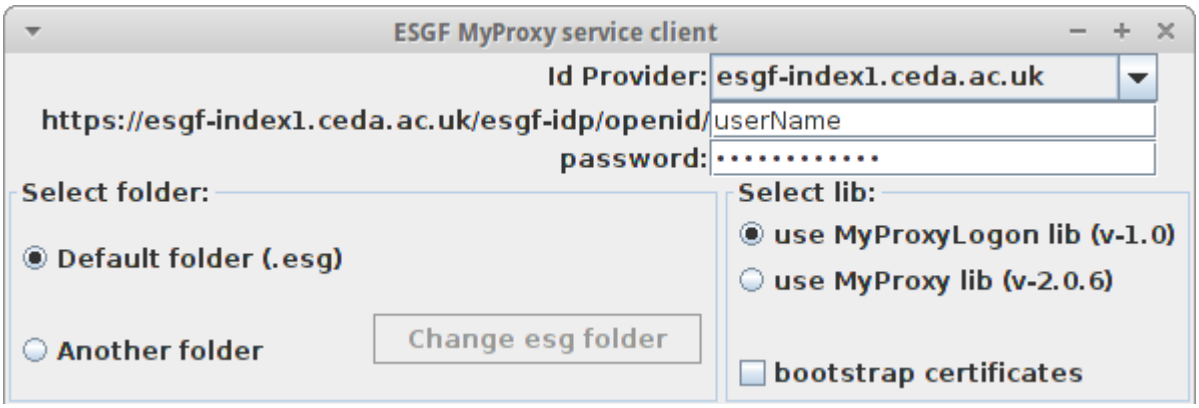
Advanced options

1. You can `bootstrap` the certificates. For that, select the check box "bootstrap certificates" in "Select Lib" section
1. You can change the output folder. The default is $USER_HOME/.esg



1. You can download a multilib myproxy version to select it in the "Select Lib" section
   - MyProxyLogon lib v1.0
   - MyProxy lib v2.0.6

   MultiLib jar -> getESGFCredentialsMultLib-0.1.jar 2.3 MB new



Use cases

Some enviroment varibles can be set:

1. The path where the user's credentials and ESGF peers certificates will be retrived

```
ESGF_HOME=.esg
```

2. The user's OpenId and password

```
OPENID=https://esgf-data.dkrz.de/esgf-idp/openid/testuser
OPENID_PASS=userpassword
```

For convenience the user's credentials and trust certificates will be retrived in JKS and PEM formats:

```
java -jar getESGFCredentials.jar --openid $OPENID --password $OPENID_PASS --writeall --output $ESGF_HOME
```

In the following use cases this URLs will be used:

1. A URL for HTTP file dowloading

```
HTTP_URL=http://wdcc-esgf.dkrz.de:8080/ESGF/fileServer/cmip5/output1/IPSL/IPSL-CM5A-LR/esmrcp85/6hr/atmos/6hrPlev/r1i1p
```

2. A URL for DODS/OPeNDAP access

```
DODS_URL=http://esgf-data1.ceda.ac.uk/thredds/dodsC/esg_dataroot/cmip5/output1/IPSL/IPSL-CM5A-LR/esmrcp85/6hr/atmos/6hr
```

Aria2

?aria2 is a lightweight multi-protocol & multi-source command-line download utility. It supports HTTP/HTTPS, FTP, BitTorrent? and Metalink. aria2 can be manipulated via built-in JSON-RPC and XML-RPC interfaces.

File download

```
aria2c --private-key=$ESGF_HOME/credentials.pem --certificate=$ESGF_HOME/credentials.pem --check-certificate=true --ca-cer
```

Retriving files from Metalink

1. Get a metalink of ESGF Files
   - Download this metalink file -> example_metalink
     For more info, ESGFToolsUI generates metalinks of ESGF files: https://meteo.unican.es/trac/wiki/ESGFToolsUI#ExporttoMetalink
1. Retrieve ESGF credentials in $HOME/.esg

```
java -jar getESGFCredentials-0.1.jar --openid <openid> --password <password> --credentials --cacertspem
```

Run aria2c with credentials and **example_metalink**

```
aria2c --private-key=$USER_HOME/.esg/credentials.pem --certificate=$HOME/.esg/credentials.pem --check-certificate=true
```

cURL

File download

```
curl --location --continue-at - --cookie curl-cookie --cert $ESGF_HOME/credentials.pem --cacert $ESGF_HOME/ca-certificates
```

- Explanation of cURL options:
  - **-L** (`L/--location`) If the server reports that the requested page has a different location let curl attempt to reattempt the get on the new place
  - **-C <offset>** (`-C/--continue-at`) to ontinue/Resume a previous file transfer at the given offset. "-C -" is used to tell curl to automatically find out where/how to resume the transfer.
  - **--cookie-jar <cookie-name>** (`-c/--cookie-jar`) to write cookies (cookies are generated after esgf-orp)
  - **--cookie <cookie-name>** (`-b/--cookie`) to load cookies from file
  - **--cert <certfile>** (`-E/--cert`) to use the specified certificate file when getting a file with HTTPS. The certificate must be in PEM format. Certificate file must content user certificate and private key.
  - **--cacert <cacertfile>** to use the specified certificate file to verify the peer. The file may contain multiple CA certificates. The certificate(s) must be in PEM format.
  - **-O** (`-O/--remote-name`) to write output to a local file named like the remote file we get. You can use (`-o/--output` <file-name> option) to specify the name of the file.

GNU Wget

NetCDF-C

The NetCDF-C libray from version 4.1 can be compiled with DAP support. Check with `nc-config` command if your NetCDF libray has been compiled with DAP support. See ?https://www.unidata.ucar.edu/software/netcdf/docs/netcdf/DAP-Support.html.

DAP access is based on ?libcurl library. The configuration paramenters are based on a file named `.dodsrc` existing in the current working directory or user's home

```
echo -e ' HTTP.SSL.VALIDATE=1 \n HTTP.SSL.CAPATH=$ESGF_HOME/certificates \n HTTP.SSL.CERTIFICATE=$ESGF_HOME/credentials.pe
```

NetCDF-Java

The NetCDF-Java library can use the credentials and trust store by defining JVM properties as command line arguments:

```
NCJ_PROP=-Dkeystore=$ESGF_HOME/keystore_jks.ks -Dkeystorepassword=changeit -Dtruststore=$ESGF_HOME/esg-truststore.ts -Dtru
```

For more info visit ?http://www.unidata.ucar.edu/software/thredds/current/netcdf-java

NCdumpW

Dump DODS/OPeNDAP URL metadata:

```
java $NCJ_PROP -cp netcdf-java/toolsUI-4.3.jar ucar.nc2.NCdumpW $DODS_URL -cdl
```

ToolsUI

Open a dataset (i.e. DODS/OPeNDAP) with NetCDF-Java's ToolsUI Java application:

```
java $NCJ -jar netcdf-java/toolsUI-4.3.jar $DODS_URL
```

ESGF WGET Script (Linux)

```
java -jar getESGFCredentials-0.0.2.jar -o <openid> -p <password> --credentials --cacerts --cacertsjks
```

ESGF WGET Script (cygwin)

```
java -jar getESGFCredentials-0.0.2.jar -o <openid> -p <password> --credentials --cacertspem --cacertsjks
```

Developers Guide

Github

?https://github.com/SantanderMetGroup/esgf-getcredentials

Architecture

See Also

- ESGFToolsUI - a desktop client for ?ESGF services