L3 Bioinformatics Limited
Life, Logic and Legend

GUIDE

Version: Draft v3, 22 January 2015

BGI Online Command Line Interface User Guide

1. Introduction

The BGI Online command line interface (CLI) is a command line program written in Python. It enables users and administrators of BGI Online to perform a set of operations against the BGI Online system. Each supported operation can be invoked as a command in the program. While each operation is performed independently, the users and administrators can mix and orchestrate their workflow by writing their custom scripts to chain up the commands. Essentially, the users and administrators can create the users accounts and projects needed, and then upload the data from upstream systems to BGI Online, and even start analysis jobs for the uploaded data, using the command line interface.

The command line interface enables automation of workflow and integration of BGI Online with BGI's production systems. Information inheritance (e.g. from LIMS) can be made possible via setting of metadata to the uploaded file too.

2. Installation and Configuration

2.1. Pre-Requisite

The BGI Online CLI is developed using Python 2.7. It also depends on the json Python package. The machine running the BGI Online CLI should have these software pre-installed. Their installation procedures are outside the scope of this document.

2.2. Installation

To install BGI Online CLI, just unpack the package into the installation folder:

```
    $ mkdir -p /path/to/install/cli; cd /path/to/install/cli
    $ tar xzvf bgi-online-cli.tar.gz
    $ ln -s /path/to/install/cli/bgionline.py /usr/local/bin/bgionline
      (super user privilege needed, optional)
```

2.3. Configuration

The configuration of BGI Online CLI can be tweaked by setting environment variables. The following table shows the configuration options.

Environment Variable	Meaning	Default Value
BGIONLINE_SERVER_HOST	The FQDN or IP of the BGI Online API Server	www.bgionline.com
BGIONLINE_SERVER_PORT	The HTTP port number of BGI Online API Server	8080
BGIONLINE_SERVER_SECURE_PORT	The HTTPS port number of BGI Online API Server	8081
BGIONLINE_SERVER_USE_HTTP	Whether to force CLI to use non-secure HTTP mode to connect to BGI Online API Server	False

BGIONLINE_TSUNAMID_PATH	The absolute path of the tsunamid uploader (e.g. /home/user/tsunamid)	Current working directory

3. Command Line Functions

The supported command of the BGI Online CLI can be seen by typing the bgionline command with the -h option:

```
    $ bgionline -h
```

It will return the list of supported commands:

```
    Usage:
    bgionline <command> [options]

        Valid commands:
            add_user, add_project, change_user_status, clone_app_to_project
            delete_file, get_download_link, list_project_files
            list_project_jobs, list_public_apps, list_job_files
            list_user_projects, login, query_job_status, set_file_metadata
            submit_job, upload_file
    bgionline <command> -h                  print help for command
    bgionline -h                            print this message
    bgionline -v                            print version

    $
```

The usage of each of the commands supported can be checked by their own -h option. For example, for the login command, you can invoke its online help by running:

```
    $ bgionline login -h
```

Please refer to the online help and this documentation for detailed usage.

The rest of this section will give details of each of the supported commands.

3.1. Login

Login is required before the user / administrator can execute any commands in the CLI. To login, use:

```
    $ bgionline login
```

It will prompt for the user name and password interactively.

```
    $ python login.py
    Username: <type in the user name>
    Password: <type in the password (not displayed)>
    Logging into the system...

    Login Success
    Token written to file: token.key
    $
```

This command also supports non-interactive mode:

```
    $ bgionline login -u <user> -p <password>
```

If the login is successful, a file will be generated under the current directory for holding the current logged in user name and the access token.

It will be read by the subsequent commands, so that users need not supply user name and password upon each command. By default, the access token file will be named token.key. The file name can be explicitly specified via the -o option. If this file is removed or the login session expired, the user / administrator will need to perform the login action again using the login command.

By default the other commands will look for the token file named token.key for the access token of the current logged in user. One can explicitly specify another token file via the -i option (when calling the other commands).

3.1.1. Daemon Mode

As an option, the login command can turn itself into a daemon after successful login. As such, the daemon will periodically check the expiry date of the current token. It will refresh the login token before it is expired. To use the daemon option, add a -d option in the command. To stop the daemon, use -s option. The -s option will not do any login action; it will only try to kill any login daemon started previously.

3.2. Add User

The add_user command creates a user account at the BGI Online system.

```
    $ bgionline add_user [options] <user> <role>
```

This command requires the current logged in user (as stored in token file, see Login command) to be an administrator. The two mandatory parameters are the user name of the user to be created and his/her role in the system respectively. The role should be one of the three values: user, author and admin.

Optionally, one can specify the password and the email address of the newly created user via the -p and -e options respectively. If these options are absent, the CLI will generate temporary values in the system. Note that the temporary values generated would not be returned. Those values will be overridden via the transfer process, which the administrator will transfer the account to the user via the change_user_status command. In the transfer process, the ultimate owner of the user account will have a chance to input his/her email address and password to the system. And the system will verify the email address in the process.

The command will have the following output:

```
    Message
    User <username> Create Success
    $
```

3.3. Add a Project and Admin Privilege to the Project

The add_project command creates a project at the BGI Online system. The newly created project will be owned and paid by the specified user.

```
    $ bgionline add_project [options] <owner_user> <project_name>
```

This command requires the current logged in user (as stored in token file, see Login command) to be an administrator. The two mandatory parameters are the user name of the owner of the project (must exist, not necessarily the administrator himself/herself), and the name of the project respectively.

Optionally, one can specify the description of the newly created project via the -d option. Also, the administrator can add himself/herself as a member of the newly created project via the -a option. In this case, the administrator will be a member of the project with view, upload file and run job privileges.

The command will have the following output:

```
    id  name
    0a10312a-5ad2-4b9e-9d90-b2068452fd5b    <project name>
```

The id field in the output can be used in other commands to identify this newly created project.

3.4. Change User Status

The change_user_status command activates/inactivates a user in the BGI Online system. It can also let the administrator transfer the account to the user.

```
    $ bgionline change_user_status [options] <user> <status>
```

This command requires the current logged in user (as stored in token file, see Login command) to be an administrator. The two mandatory parameters are the user name of the user whose status is to be changed and the target status respectively. The status should be one of the three values: activate, inactivate and transfer.

The activate and inactivate commands would activate and inactivate the user in the BGI Online system respectively. For the transfer command, the system will change the user account into the transfer state, and generate a transfer token.

The transfer command will have the following output:

```
    $ bgionline change_user_status user1 transfer
    token   expireAt
    KsMRgXbPpEKkt9zbYuCeAdxvk4TEdVU1    2014-12-07+12:42:16
```

Under this transfer state, the BGI Online system will accept the user to change his/her email address and new password via a special account transfer page. To do so, the administrator can send a link, using his/her own means, with the generated transfer token to the user and let him/her finish the transfer process. The link to do so is:

```
    https://<bgi online server host name>/#/account/transfer?username=<usern
ame>&token=<token>
```

When the user clicks on the link, he/she will be presented with a page on which he/she can input his/her real email address and real password, and this will complete the transfer process.

3.5. Clone App To Project

The clone_app_to_project command adds an app to a project.

```
    $ bgionline clone_app_to_project [options] <app_id> <project_id>
```

This command can be run by any user having the adequate privilege in the specified project. The two mandatory parameters are the ID of the app to be cloned and the ID of the project to hold the cloned app respectively. Optionally, one can specify the name of the newly cloned app via the -n option. Also, one can specify the version string of the newly cloned app via the -v option. If these options are missing, the command will copy the name or version string from the source app to the cloned app.

The command will have the following output:

```
    id
    1fe7bee0-9259-11e4-96fa-123b93f75cba
```

The id field is the ID of the newly cloned app.

3.6. Get Download Link

The get_download_link command gets the link for downloading a file in the BGI Online system.

```
    $ bgionline get_download_link [options] <file_id>
```

This command can be run by any user having the adequate privilege in the specified project. The only mandatory parameter is the ID of the file to be downloaded.

This command will have the following output:

```
    url
    <the_url_for_downloading_the_file>
```

The link is accessible for downloading the file using HTTPS protocol.

3.7. List Project Files

The list_project_files command gets all the files under the specified project.

```
    $ bgionline list_project_files [options] <project_id>
```

This command can be run by any user having the adequate privilege in the specified project. The only mandatory parameter is the ID of the project to be listed.

The command will have the following output:

```
    id  size    name    meta
    8dfaab33-0001-4aa7-b646-597c929943ec    111.5 MB    human_hg19_genes.gtf
       "File type"="gtf", "Lane"="4", "Sample ID"="hg19.raw"
    f18c698f-cd1b-4c4c-bf76-cbc2e0dc4d96    94.9 MB
example_human_Illumina.sorted.zip"File type"="zip", "Lane"="4", "Sample
ID"="hg19.raw"
    ...
```

Each row, except the header row, represents the information of a file under the project. The information is presented in a tab-delimited format. The last part of the information is the metadata of the file in the system, which is a list of key-value pairs.

3.8. List Project Jobs

The list_project_jobs command gets all the jobs under the specified project.

```
    $ bgionline list_project_jobs [options] <project_id>
```

This command can be run by any user having the adequate privilege in the specified project. The only mandatory parameter is the ID of the project to be listed.

The command will have the following output:

```
    id  name    status
    bf5f15f3-0005-49c9-8ee2-aba45b32015b    FastQ Split-Merge run - 28-08-
2014 09:17:42 Finished
    f18c698f-cd1b-4c4c-bf76-cbc2e0dc4d96    JobName423434   Finished
    c349e315-eb08-406c-8b6d-3e1c13ac9d39    Split Read Alignment for RNA-Seq
- TopHat   Exception
    ...
```

Each row, except the header row, represents the information of a job under the project. The information is presented in a tab-delimited format.

3.9. List Public Apps

The `list_public_apps` command gets all the publicly shared apps in the BGI Online system.

```
    $ bgionline list_public_apps [options]
```

This command can be run by any user having the adequate privilege in the system. No mandatory parameter is required.

The command will have the following output:

```
    id  name    version
    346fff0b-0001-4475-9167-01414f42b978    Exome Variant Calling - BWA +
GATK  1.0
    f23c0544-6791-4d03-a3e1-b527dc372c7a    FastQC Analysis 1.0
    5b0e5685-b7d6-4cf9-be32-6a2dc8a5412a    Grab File from URL  1.0
    ...
```

Each row, except the header row, represents the information of a public app in the BGI Online system. The information is presented in a tab-delimited format.

3.10. List Job Files

The `list_job_files` command gets all the files generated by the specified job.

```
    $ bgionline list_job_files [options] <job_id>
```

This command can be run by any user having the adequate privilege in the relevant project. The only mandatory parameter is the ID of the job to be checked.

The command will have the following output:

```
    id  size    name    meta
    8dfaab33-000a-4aa7-b646-597c929943ec    999.038 KB  merged.fastq
"File type"="fastq"
    ...
```

Each row, except the header row, represents the information of a file under the project. The information is presented in a tab-delimited format. The last part of the information is the metadata of the file in the system, which is a list of key-value pairs.

3.11. List User Projects

The `list_user_projects` command gets all the files generated by the specified or currently logged in user.

```
    $ bgionline list_user_projects [options] [<user_id>]
```

This command can be run by any user in the BGI Online system. The command may take an optional parameter, which is the ID of the user to be checked. If this optional parameter is not specified, the command will list the projects of the currently logged in user.

The command will have the following output:

```
    id  name
    026ba7db-0001-45fa-9766-74a9929c552a    My First Project
    c7aa7e6a-8b56-405e-9f52-ca61c4f6d0fb    Test project 2
    6501a643-b964-4fde-864d-5f8568a551a3    Test project 3
    ...
```

Each row, except the header row, represents the information of a project, which the user being checked is a member. The information is presented in a tab-delimited format.

3.12. Query Job Status

The `query_job_status` command queries the status of the specified job in the system.

```
$ bgionline query_job_status [options] <job_id>
```

This command can be run by any user having the adequate privilege in the relevant project. The only mandatory parameter is the ID of the job to be checked.

The command will have the following output:

```
    name    status  startTime   stopTime
    FastQ Split-Merge run - 28-08-2014 09:17:42 Finished    2014-08-28
09:04:23 2014-08-28 09:06:24
```

The command shows the name, status, start time and stop time (if any) of the job being checked.

3.13. Submit Job

The `submit_job` command submits an analysis job to the system.

```
    $ bgionline submit_job [options] <job_name> <project_id> <app_id>
[<job_config_file>]
```

This command can be run by any user having the adequate privilege in the relevant project. The three mandatory parameters are the name of the new job, the ID of the project in which the job will be started, the ID of the app of the job respectively. Optionally, one more parameter of the path of a configuration file can be specified. If the file path of the configuration file is not specified, the system will use an empty configuration for starting the job.

Optionally, one can specify a description for the newly created job via the `-d` option.

The command will have the following output:

```
    id
    11601c20-9261-11e4-96fa-123b93f75cba
```

The `id` field is the ID of the newly created job.

3.14. Upload File

The upload_file command uploads a file to the specified project. To upload a file, the command line tool will invoke another program called tsunamid to perform the actual uploading work. The path of the tsunamid program can be specified by the BGIONLINE_TSUNAMID_PATH environment variable. If this environment variable is not set, the command line tool will look for the tsunamid program in the current working directory.

The tsunamid program needs a specified port number for network communication. Important note: the firewall should be configured properly so that Internet traffic (using both TCP and UDP on the specified port) should be able to be sent to the computer running the upload program.

To invoke the upload task:

```
    $ bgionline upload_file [options] <project_id> <file_path_to_upload>
```

This command can be run by any user having the adequate privilege in the relevant project. The two mandatory parameters are the ID of the project in which the uploaded file will be contained, and the path of the file to be uploaded respectively.

Optionally, one can specify metadata to the file via the -m option. Note that the -m option can be specified multiple times to specify multiple metadata to associate with the file to be uploaded. The program will also detect the name of file to be uploaded. If it is in a specific format, it will parse it and fill in metadata for the file automatically.

For example, if the file is named

```
    110114_I481_FC81C7HABXX_L5_HUMiqvDBTDIAAPE_1.fq.gz
```

It will generate metadata as follows:

```
    Year = 11
    Month = 01
    Day = 14
    Flowcell = FC81C7HABXX
    Lane = 5
    Library = HUMiqvDBTDIAAPE
    Pair = 1
```

Also, one can specify the network port for communication via the -p option; and specify the network level via the -n option (1, 2, 3, 4, 5 map to 10, 50, 100, 500, 1000Mbps in maximum, respectively). To let the command output more information while the uploading program is doing its tasks, one can turn on the verbose mode via the -v option.

The command will print the return code of the uploading program:

```
    Uploader returned 0
```

If the program exits with return code 0 (like the example above), the upload process is successful. Otherwise, the upload process has failed. Please refer to the verbose message (available in verbose mode) and investigate.