# CLC **Server Command Line Tools**

USER MANUAL

Manual for

# *CLC Server Command Line Tools 5.0.1*

Windows, macOS and Linux

March 8, 2018

**This software is for research purposes only.**

QIAGEN Aarhus
Silkeborgvej 2
Prismet
DK-8000 Aarhus C
Denmark

# Contents

# Chapter 1

# Introduction

Welcome to the user manual of *CLC Server Command Line Tools 5.0.1*.

The *CLC Server Command Line Tools* is a command-line client for the CLC Server solutions[1]. The *CLC Server Command Line Tools* provide the tools to start analyses and other tasks on CLC Servers, including data import and export, and utility data operations such as moving, renaming, and deleting data on the server.

A typical work flow using the *CLC Server Command Line Tools* might be:

1. Import your sequence or structure data
2. Run analyses such as read mapping, SNP detection, RNA-Seq, or Docking Ligands
3. (Optionally) export the results to your local disk

Other clients available to run tasks on the *CLC Server* are the graphical CLC Workbenches. Below are recommendations for choosing which of these two types clients, the graphical or the command line, to use for your work:

- For *visualization and interpretation of data* we recommend using a CLC Workbench. If results are generated using the *CLC Server Command Line Tools*, then these can be viewed using a CLC Workbench that is connected to the same CLC Server the analyses were carried out on. Alternatively, the data can be exported and shared.
- For *explorative work* we recommend using a CLC Workbench. The effects of parameter changes, for example, are easier to interpret using the graphical interface. For many users, selection and management of data is also more intuitive through a graphical interface. In addition, the graphical user interface has more constraints to help guide reasonable choices of parameters and combination of parameters; these constraints are not all present in the *CLC Server Command Line Tools*.
- Automation and consistency in production environments can be supported by using the *CLC Server Command Line Tools*. In particular, you can *script pipelines* of analyses on the *CLC*

[1] Like any other client software, the *CLC Server Command Line Tools* would most commonly be installed and used on systems other than the one that the CLC Server software is installed on, although there is no restriction meaning that this must be the case.

*Server*, and then use these scripts for processing many data sets in a consistent manner. Consistency is also supported in the CLC Workbenches by executing Workflow analyses in batch runs, but automation is still best supported through use of the *CLC Server Command Line Tools*.

This user manual begins with installation instructions followed by an explanation of the basics of operating the *CLC Server Command Line Tools* and usage information.

## 1.1 Installation

The *CLC Server Command Line Tools* can be downloaded from http://www.qiagenbioinformatics.com/products/clc-server-command-line-tools/ and is available for Windows, Mac and Linux. You can install the tools on any computer that can connect to your *CLC Server*.

The system requirements of *CLC Server Command Line Tools* are these:

- Windows 7, Windows 8, Windows 10 or Windows Server 2012
- OS X 10.9, 10.10, 10.11 and macOS 10.12
- Linux: RHEL 6.0 and later, SUSE 13.1 and later.
- 64 bit
- 1 GB RAM required
- 2 GB RAM recommended
- 1024 x 768 display required
- 1600 x 1200 display recommended

You will also need a running version of *CLC Server*. No additional license is required for running the *CLC Server Command Line Tools*.

# Chapter 2

# Basic usage

Once installed, there will be four programs present in the installation folder:

- `clcserver` - the key program. It is used to run all the commands that communicate with the server.
- `clcresultparser` - used to parse data locations from particular text files generated during clcserver runs. This command is most useful when connecting analyses in a scripting pipeline (see section 2.4).
- `clcserverkeystore` - a helper tool for enabling passwords to be handled securely (see section 2.1).
- `clcserversslstore` - a helper tool for managing SSL certificates (see section 2.2)

The `clcserver` program requires the following four flags, which provide information about the connection to the server:

**–S <hostname or IP address of the server>**

**–P <port the server runs on>** When omitted, port 7777 is used, which is the default for server installations.

**–U <user name>** The username used to log into the server.

**–W <password or token>** See section 2.1 for how to avoid entering passwords in clear text.

If you run the clcserver command with the above parameters, and nothing else, then a list of all commands that can be run on the server will be returned. For example:

```
clcserver –S server.com –U bob –W secret
```

The commands to be run on the server are supplied with the flag:

**–A <command to be executed on server>**

If you supply the -A flag with a program name, but do not provide the required flags for that program, then a listing of the flags for that program will be returned. For example, a command of a form like:

```
clcserver -S server.com -U bob -W secret -A read_mapping
```

would return the full list of parameters for the `read_mapping` function, including the possible values, and descriptions. This information, for each command, is also available in the online manual at http://resources.qiagenbioinformatics.com/manuals/clcservercommandlinetools/current/ in the "Usage for *CLC Server name*" chapters.

An optional flag when working on the command line, but important when working with scripts, is:

**-O <filename>** The name of a file to be created to hold a summary of steps carried out on the server and data locations of the results generated. The data locations are of a form that can be used by downstream CLC commands. See section 2.4 for information about parsing this file. By default, this file is placed in your working directory. If you do not provide this flag, this data will be written to a file called `results.txt`

For those working with the *CLC Grid Integration Tool*, you can run import and algorithm commands through your grid nodes by adding the following flag to your clcserver command:

**-G <grid preset name>**

Other optional flags available for the clcserver command are:

**-C <integer>** Specify the **column width** of the help output.

**-D <boolean>** Enables **debug** mode when set to true, providing more elaborate output and error messages.

**-H** Display general **help** instructions.

**-V** Display the **version number** of *CLC Server Command Line Tools*.

## 2.1 Handling passwords

To help you avoid sending your server login password in clear text across the network, we provide the `clcserverkeystore` tool. This enables you to convert your password to a token, which is stored and can be interpreted by the *CLC Server Command Line Tools* when logging onto the server. The token is encrypted and saved with the user profile on the computer running the *CLC Server Command Line Tools*.

You can generate a password token using the following command:

```
clcserverkeystore --generate
```

You will be prompted for the password. After you have typed the password, press the **Enter** key. The password token is then returned on screen. It will be a long string of text that you should save somewhere to refer to for future use.

So, if we say that user `bob` has password `secret`, and has generated a password token `CAIHMAAAAAAAAAPcb769377f4`, then he could enter either of the following two commands to connect to his server. The first passes the password in plain text. The second, passes it as an encrypted token.

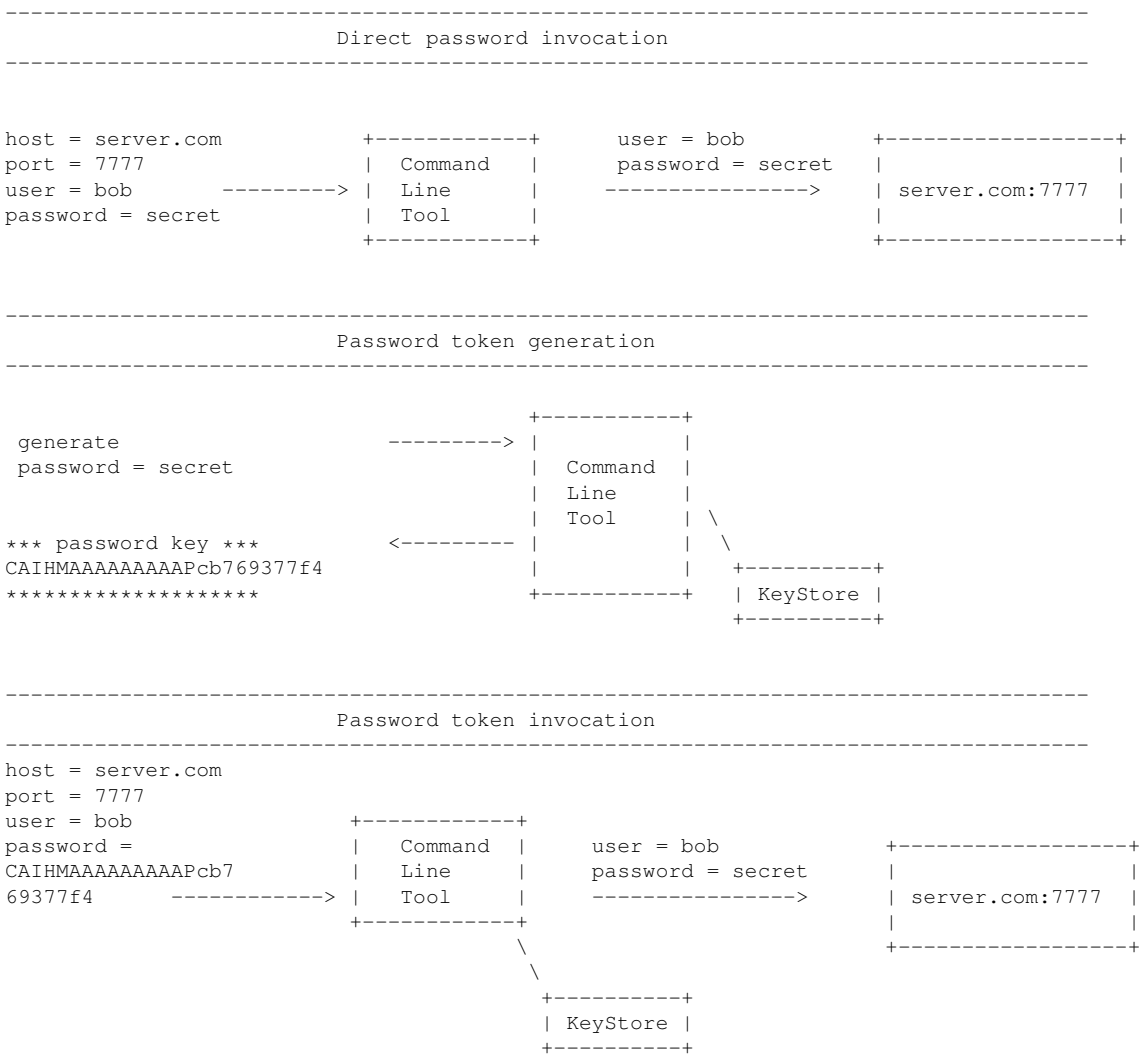
```
clcserver -S server.com -U bob -W secret

clcserver -S server.com -U bob -W CAIHMAAAAAAAAAPcb769377f4
```

If the token needs to be deleted, the `clcserverkeystore` program has two other parameters that can be used:

**-d <token>** This will delete the individual token provided as a parameter.

**deleteAll** This will delete all the tokens in the user profile.

The first section of the diagram below illustrates the process of logging into the server using a clear text password. The second section illustrates the process of generating a password token and storing it in the keystore, followed by a section showing how the token is substituted by the *CLC Server Command Line Tools* with the real password when initiating the connection to the server.



## 2.2 Managing SSL certificates

The `clcserver` command will automatically detect and use SSL if present on the port it connects to. However, if the certificate is untrusted it will refuse to login. In order to connect to a server, its certificate must be added to the trust-store by using the `clcserversslstore` utility.

When invoking `clcserversslstore` it is possible to both list and add new certificates to the trust-store. Certificates are added by providing the program with the connection information (via the -S, -P, -U, and -W parameters):

```
clcserversslstore –S server.com –U bob –W secret –P 7778
```

If the port connected to is indeed an SSL-enabled port, the program will ask if the certificate should be trusted for future `clcserver` invocation:

```
The server (server.com) presented an untrusted certificate with the
following attributes:

SUBJECT
=======
Common Name         : server.com
Alternative Names   : N/A
Organizational Unit: Enterprise
Organization        : CLC Bio
Locality            : Aarhus N.
State               : N/A
Country             : DK

ISSUER
=======
Common Name         : server.com
Organizational Unit: Enterprise
Organization        : CLC Bio
Locality            : Aarhus N.
State               : N/A
Country             : DK

FINGERPRINTS
============
SHA-1               : A5 F6 8D C4 F6 F3 C2 44
SHA-256             : 49 B5 0B 04 3C 3A A1 E2 D1 BF 87 10

VALIDITY PERIOD
===============
Valid From          : Sep 1, 2011
Valid To            : Aug 31, 2012
Trust this certificate? [yn]
```

Answering `y` to this will record the certificate in the trust-store, and allow subsequent `clcserver` invocation to connect to the server.

It is possible to list the trusted certificates by invoking the `clcserversslstore` program with the -L argument.

## 2.3 Data objects, data files and the CLC URL

In this section, we refer to data already in a CLC data area as being in a `persistence model`. This technical term allows us to refer to any area that the CLC Servers or Workbenches recognize as CLC data areas. Most relevant to this document are CLC Server File Locations and Database Locations. A given Server File Location is a single persistence model.

Whether data being referred to is in a persistence model, a designated Import/Export area or on the local filesystem of the client machine is indicated by the type of location information provided in the command:

- Data held in a CLC Server persistence model is specified using a CLC URL. Two different forms can be used, one is based on the data element name, the other uses the element's object id. These forms are described in the section below.
- Data held in an area configured as an Import/Export area on the server are specified using a CLC URL. Here, only the name form can be used.
- Files on the local, client system can be specified by giving the full or relative path to that file. This is only possible on systems where direct transfer from the local system has been allowed by the CLC Server admin [1].

### 2.3.1 The CLC URL - the ID form

Data resources within persistence models can be referred to using the object-ID form of CLC URLs. These look soomething like the following:

clc://node04:7777/3123-2131uafda-sads/213-sddsa123-5232

**Getting the object ID form of a CLC URL** There are several ways this can be done:

1. Via the Workbench.

   Copy the CLC URL by highlighting the data object by clicking on the object in the Workbench **Navigation Area** to select it, and then using the keyboard short cut Ctrl-C. Then use Ctrl-V to paste the URL into a shell window, text editor or similar (figure 2.1).

2. Using the CLC Server web administrative interface.

   Select a data object from the tree browser on the left hand side of the browser window, and then select the "Element info" tab in the main area of the browser window. Click on the link to CLC-URL. This shows two versions of the CLC URL one using the name and one using the object ID.

[1] From the CLC Genomics Server 8.0, server admins are able to specify whether direct data transfer from a local machine is allowed. If it is not allowed, you need to move the relevant files to an area configured as an Import/Export area on your CLC Server.

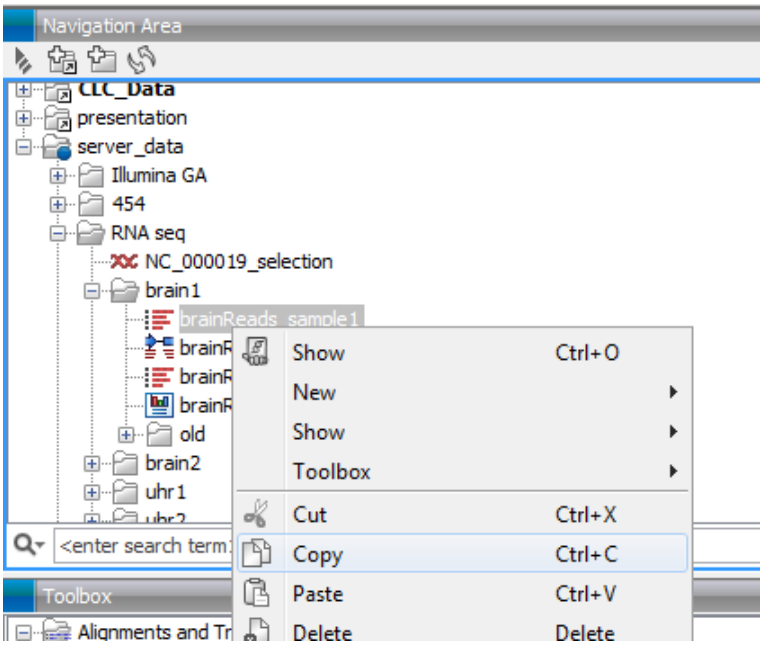



Figure 2.1: *Copying a data object in the workbench will put the CLC URL on the clipboard. You can then paste the URL into your command in the terminal.*

3. Take the object ID from within the text ouptut file generated using the -O flag of the `clcserver` command.

   This would be the common route when running a series of commands via a script.

**Benefits of the ID form:** The ID form of a CLC URL is impervious to changes to the name of a data object or the folders the data resides in. That is, such changes do not affect a data object's ID.

**Drawbacks of the ID form:** The ID form is not directly interpretable by humans.

### 2.3.2 The CLC URL - name form

The object-name form of CLC URLs can be used to refer to data resources within persistence models or to refer to files located on the machine the CLC Server software is running on.

The first section of a object-name form of a CLC URL indicates whether it is referring to a **data object** in a CLC Server persistence model or to a **file** stored in an area configured as an Import/Export location for the CLC Server. These forms are:

**clc://server** clc://server URLs refer to a data object present in a persistence model. This part of the URL is then followed by the name of the persistence model the data is located in. For example, the name of particular CLC Server File Location. An example of this form is: `clc://server/CLC_Server_Project/alignments/myAlignment`

Note that for CLC data in a persistence model, you need the **name of the data object** (as seen via the **Navigation Area** of the Workbench) **not** the name of the file holding the data object (as seen when listing files using system tools like the commands ls or dir).

**clc://serverfile** clc://serverfile URLs refer to a file in an area configured as an Import/Export area for the CLC Server. This form would commonly be used to point at files containing data that is about to be imported into the CLC Server, or to indicate a location to export data to. An example of this form is:

```
clc://serverfile/mnt/data/project1/s_1_1.sequence.txt
```

**Benefits of the name form:** Human readable and easier for many people when first starting out working with the Command Line Tools or when just running a few commands directly (as opposed to via a script).

**Drawbacks of the name form:** Any changes to the names of data objects or folders in the persistence model will break the URL.

### 2.3.3 Indicating local system files or folders

When importing files on the local system (i.e. the machine the `clcserver` command is being run on), or exporting to the local system, then the *relative or full path* should be specified in the command. An example would be:

```
/home/username/somefolder/datafile.gb
```

From the CLC Genomics Server 8.0, server admins can specify if direct import from a local machine to the CLC Server system is allowed. If it is not, data to be imported must be transferred to a designated Import/Export location, and a CLC URL used to indicate where the data is to be imported from.

## 2.4 Result files and connecting analyses in pipelines

For each run of `clcserver`, text information is returned providing a summary of the steps taken, and the locations, in ID form, of any files generated. The file containing this information will, by default, be created in the *current directory* and will be called `result.txt`. You can use the `-O` option for the `clcserver` command if you wish to specify an alternative file to be written to.

An example of contents in a typical result file is shown below. In this case the file that was generated after running the trim algorithm using a sequence list called `reads` as input. The result file lists the three files that were produced.

```
//
Name: reads trimmed
ClcUrl: clc://127.0.0.1:7777/-268177574-YCAAAAAAAAAAAAPc673b0db8c5e724f--5d66a991-12d75090d93--7fff
//
//
Name: reads report
ClcUrl: clc://127.0.0.1:7777/-268177574-ADAAAAAAAAAAAAPc673b0db8c5e724f--5d66a991-12d75090d93--7fff
//
//
Name: Trim Sequences log
ClcUrl: clc://127.0.0.1:7777/-268177574-CAAAAAAAAAAAAAPc673b0db8c5e724f--5d66a991-12d75090d93--7fff
//
```

When creating pipelines stitching together several analyses, you parse the result file to get the location of the data produced, which is needed as input for the next algorithm.

The result file is just a text file, but it can still be a challenge to parse it to get the necessary CLC URLs. Thus, we provide a tool called `clc_result_parser` to help with this. It searches the result file for a text expression you provide, and returns the CLC URL for files where a match to that text has been found in the `Name:` field. The `Name` field will contain the name of the input data along with a description of the type of data held in that file location.

In the case above, you would probably search for the trimmed reads to use for further analysis, which could be done with a command like this:

```
clcresultparser -f result.txt -c trimmed
```

Here, the following text would be returned:

```
clc://127.0.0.1:7777/-268177574-YCAAAAAAAAAAAAPc673b0db8c5e724f--5d66a991-12d75090d93--7fff
```

The options for the `clcresultparser` program are:

**–f <name of result file to parse>** This option is required.

**–c <text to search for>** Text to search for in the `Name` field of the result file. If nothing is found, the exit code is 1.

**–n <text that should not match>** Text that should **not be contained** in the `Name` field of the result file.

**–r <regexp>** A **Java regular expression** used for matching the name of the output (see http://java.sun.com/docs/books/tutorial/essential/regex/index.html).

**– –ignorelogs <boolean>** By default, all analyses produce log files. You can provide `false` as the argument to this option to stop log files from being returned. This is equivalent to excluding all names ending with `log`, or `log` with a number suffix. The latter are generated when there is more than one log file in the same folder.

**–p <prefix text>** When more than one match is found, the data locations for all matches will be output as a space-separated list. By supplying a **prefix** string, you can stipulate what character(s) to separate the list using. E.g. If you need to send several files output by the `clcresultparser` command as arguments to `-i` options for the next analysis, simply provide `"-i"`as as the argument for the `-p` flag.

**–e <integer>** The number of CLC URLs that are **expected** to be returned. If this is not the number of results files that match the search string. the command will return with exit code 10. This option is designed for use in scripts where you will wish to carry out validation steps are you proceed through the pipeline. (On the command line, you check the error code returned by the previous command by typing `echo $?`.

**–C <integer>** Specifies the **column width** of the help output.

## 2.5 Executing workflows

It is possible to execute workflows installed on the server. Workflows are described in detail in the user manuals of the CLC workbenches and *CLC Server* at http://www.qiagenbioinformatics.com/support/manuals/.

Executing workflows is similar to executing algorithms, and the installed workflows will be listed when the `-A` is omitted. Parameters that are open for change on execution are displayed when the workflow is specified for the `-A` option. Please note that the parameter names have name of the workflow element pre-pended to make sure they are always unique.

## 2.6 Emptying the recycling bin for a CLC Server File Location

Each CLC Server File Location has a recycling bin, where files that users delete are put. Only members of the administrator group, as defined on the CLC Server, can empty the recycle bin associated with CLC Server file locations. This is because the recycle bin is a shared location for any given CLC Server file location and many sites do not want all users to be able to access it directly, that is to be able to view things or delete other people's data.

One can avoid the need to periodically go in and manually empty recycle bins by setting up a script that is run as a cronjob, which includes a command of the following form:

```
clcserver -S <serverinfo> -P <portnumber> -U <adminusername> -W <password or token> -A empty_recycle_bin -t clc://server/$LOCATIONNAME
```

Figure 2.2: *How to set up a script that automatically empties the recycle bin.*

Above, `$LOCATIONNAME` would be replaced by the name of the CLC Server File Location you wish to empty the recycling bin of.

# Chapter 3

# Usage for all commands

Usage information for all commands in the latest version of the *CLC Server Command Line Tools* can be found in the "Usage for *CLC Server name*" chapters of the html manual: http://resources.qiagenbioinformatics.com/manuals/clcservercommandlinetools/current/.

For the *CLC Server Command Line Tools* 2.2.1 to the present, documentation for version can be found by substituting the version number of the software without any full stops (periods) in place of "current" in the URL above. For example for version 3.5.1, the URL would be: http://resources.qiagenbioinformatics.com/manuals/clcservercommandlinetools/351/.

Commands for versions of the *CLC Server Command Line Tools* 1.6.1 through 2.2.0 inclusive can be found here: https://www.qiagenbioinformatics.com/clc-server-command-line-tools-legacy-documentation/.