UNIC◉RE

UFTP STANDALONE CLIENT

UNICORE Team

Document Version:	1.0.0
Component Version:	0.7.0
Date:	19 07 2017

PDF BY DBLATEX

Contents

This is a Java-based client for UFTP. It allows to

- list remote directories

- upload/download files

- sync files

- make remote directories

- delete remote files or directories

- manage shares and access shared data

The *uftp* client will connect to a UFTP Authentication Server to authenticate and then to the *uftpd* server for transferring data or making a file operation.

The *uftp* client supports username/password authentication, OIDC token authentication and (on UNIX) ssh-key authentication.

Prerequisites

- Java 8 (OpenJDK, Oracle, IBM)

- Access to a UFTP authentication service and the corresponding UFTPD server. To use the client, you need to know the address of the authentication service.

- You need valid credentials for the UFTP authentication service

- If the UFTP authentication service is configured to use SSH-key authentication, you cannot use this client on Windows. We do not yet support SSH keys on Windows.

Installation

Unzip the archive in a location of your choice. Add the *bin* directory to your path. (Alternatively, you can copy *bin/uftp* script to a directory that is already on your path, in this case edit the script and setup the required directories.)

Basic Usage

In this manual, we use the following format to indicate commands that you can type on the command line:

```
$> some_command
```

and assume that the bin directory of the UFTP client is on your path.

Invoking uftp without any arguments,

```
$> uftp
```

will list the available commands.

Invoking

```
$> uftp <command> -h'
```

will show help for a particular command

Authentication

To specify the remote username, use the "-u <username>" option, e.g.

```
$> uftp ls -u username https://localhost:9000/rest/auth/TEST:/home/ ↩
    demo/
```

The credentials can be given in multiple ways.

- directly using the "-u username:password"

```
$> uftp ls -u username:password ...
```

- tell the uftp client to query the password interactively by additionally giving the "-P" option, e.g.

```
$> uftp ls -P -u username ...
```

- If no password is given, the client will attempt to use your SSH key for authentication, this has to be configured on the authentication server accordingly.

- You can also directly specify a value for the HTTP *Authorization* header with the "-A" option. This allows to use an OIDC bearer token for authorization, e.g. -A "Bearer <oidc_to ken>". In this case no username is required.

```
$> uftp ls -A "Bearer <oidc_token>" ...
```

Examples

In the following examples, the authentication service is located at "localhost:9000/rest/auth/" and the user name is *demo*. Replace these values by the correct ones for your installation.

Listing a directory: the "ls" command

```
$> uftp ls -u username:password https://localhost:9000/rest/auth/ ↩
    TEST:/home/demo/
```

will list the */home/demo* directory.

Copying data: the "cp" command

The *cp* command is used to copy local data to a remote server or vice versa. Remote locations are indicated by the "https://" prefix, and you need your user name, and the URL of the authentication server.

It has a number of features, which will be shown in the following.

Basic usage

Downloading a single file:

```
$> uftp cp  -u username:password https://localhost:9000/rest/auth/ ↩
    TEST:/home/demo/test.data .
```

will download the */home/demo/test.data* file to the current directory

Download files using wildcards:

```
 $> uftp cp -u username:password https://localhost:9000/rest/auth/ ↩
     TEST:/home/demo/data/* .
```

will download all files in the */home/demo/test* directory to the current directory

Similar commands work for upload.

The recurse flag, *-r*, tells uftp to also copy subdirectories.

Piping data

The "cp" command can read/write from the console streams, which is great for integrating uftp into Unix pipes. The "-" is used as a special "file name" to indicate that data should be read/written using the console.

For example to tar the contents of a directory and upload the tar file using uftp:

```
$> tar cz dir/* | uftp cp -u username:password - https://localhost ↩
    :9000/rest/auth/TEST:/archive.tgz
```

Similarly, "-" can be used to write data to standard output. As an example, consider this:

```
$> uftp cp -u username https://localhost:9000/rest/auth/TEST:/ ↩
    archive.tgz - | tar tz
```

Or use uftp to cat a remote file:

```
$> uftp cp -u username https://localhost:9000/rest/auth/TEST:/foo. ↩
    txt -
```

Byte ranges

To copy just part of a file, a byte range can be given with the "-R" option. Counting starts at "zero". For example to download only the first 1024 bytes of file (i.e. the range 0 - 1023), you would do

```
$> uftp cp  -u username -R 0-1023 https://localhost:9000/rest/auth/ ↩
    TEST:/home/demo/test.data .
```

As an additional feature, you can use the additional "-p" flag, which will write also only the given range. For example

```
$> uftp cp -u username -R 1024-2047-p https://localhost:9000/rest/ ↩
    auth/TEST:/home/demo/test.data .
```

will write bytes 1024-2047 of the remote file to the local file, starting at offset 1024.

The same thing works for remote files!

This feature has a number of interesting uses, for example parallel clients reading different parts of the same remote file, multiple server/client pairs for added throughput, etc.

Encryption and compression

The cp command supports the "-E" and "-C" options, which enable data encryption and compression (during transfer) respectively. These work only if a single data stream is used.

Data encryption uses a symmetric algorithm, which nonetheless drastically lowers the performance.

Data compression uses the gzip algorithm.

Compression and encryption can be combined.

Resuming a failed transfer

If a copy command was terminated prematurely, it can be resumed using the "-R" option. If the "-R" option is present, the UFTP client will check if the target file exists, and will append only the missing data.

So if your inital copy operation

```
$> uftp cp -u username https://localhost:9000/rest/auth/TEST:/home/ ←
    demo/test.data .
```

did not finish correctly, you can resume it with

```
$> uftp cp -u username -R https://localhost:9000/rest/auth/TEST:/ ←
    home/demo/test.data .
```

Synchronizing a file: the "sync" command

Currently, sync only supports single files, i.e. no directories or wildcards! The syntax is

```
$> uftp sync -u username <master> <slave>
```

For example, to synchronize a local file with a remote "master" file:

```
$> uftp sync -u username https://localhost:9000/rest/auth/TEST:/ ←
    master.file local.file
```

To synchronize a remote file with a local "master" file:

```
$> uftp sync -u username master.file https://localhost:9000/rest/ ←
    auth/TEST:/remote.file
```

Data sharing

Data sharing enables users to create access to their datasets for other users via UFTP, even if those users do not have Unix-level access to the data.

Note
The data sharing features require additional server-side features, not all UFTP installations support data sharing. You can check if a server has the sharing feature enabled by running "uftp info"

Listing shares

```
$> uftp share --list -u username https://localhost:9000/rest/share/ ←
    TEST
```

Creating or updating a share

A share consists of a target DN, a server-side path and the permissions.

For example to share "/data/public/somefile.pdf" with the user "CN=User"

```
$> uftp share -u username https://localhost:9000/rest/share/TEST " ←
    CN=User" /data/public/somefile.pdf
```

You can use the following options to modify the default "read" access:

- `--anonymous` for anonymous access (careful!). In this case no DN is given.

- `--write` for write acces

- `--delete` to delete a share

Downloading

To download a file that is shared with you, use the "get" command

```
$> uftp get -u username https://localhost:9000/rest/share/TEST/auth ←
    :/data/public/somefile.pdf local_file.pdf
```

Currently this command does not support wildcards.

Uploading

To upload a file to a shared location (file or directory), use the "put" command

```
$> uftp put -u username data/*.pdf https://localhost:9000/rest/ ←
    share/TEST/auth:/data/public/
```

Troubleshooting

This section lists and explains common error messages and their resolution.

I get "Invalid server response 500" and "Exception.... Authentication failure"

Probably you gave a wrong username or password. Contact your site administrator if in doubt!

If using a password, make sure you give the "-P" flag.

I get "Invalid server response 405 Unable to connect to server for listing"

Check the uftp:// URL that you use. Maybe you have a typo in the "/rest/auth/<servername>" part.

Getting support

UNICORE Website: http://www.unicore.eu

Support list: unicore-support@lists.sf.net

Developer's list: unicore-devel@lists.sf.net