

# protocol-assignment-1

## ToDo:

## Important:

- UDP MTU
- Don't create packages with a size of  $(\text{bit-size} \bmod 8) \neq 0$ . It makes it hard on the receiver side to interpret those!

## Changelog:

- 29.04.2018 [Fabian] Updated protocol
- 29.04.2018 [Kilian] Added `File-Status` message
- 29.04.2018 [Fabian] Added message description and `Ping` message
- 26.04.2018 [Kilian] Added Convention, added CRC and Hash, added ability to delete Files and Folders, minor optimisations
- 20.04.2018 [Fabian] Initial commit
- 22.04.2018 [Fabian] Protocol

## Protocol:

### General field descriptions:

Type [4 Bit]:

0000 => Client-Hello-Handshake  
0001 => Server-Hello-Handshake  
0010 => File-Creation  
0011 => File-Transfer  
0100 => File-Status  
0101 => ACK  
0110 => Ping

Client ID [32 Bit]:

An unique client id generated by the server on first contact.  
E.g. a static int that gets incremented for each connected client.

Checksum [32 Bit]:

CRC32 Algorithm [Wiki Link](#)

Sequence Number [32 bit]:

Like TCP

FID Length [64 Bit]:

The length of the `FID` field in bytes.

FID [Defined in the `FID Length` field]: The relative path to the file. Includes the file name e.g.  
`folder/file.txt`

## Client-Hello-Handshake:

The initial connection message that gets send by the client.

```
0      4      20      52      56
+-----+-----+-----+-----+
| Type | Port | Checksum | UNUSED |
+-----+-----+-----+-----+
```

Port [16 Bit]:

The port on which the client listens to server messages

UNUSED [4 Bit]:

To ensure the package has  $\text{mod } 8 = 0$  size

## Server-Hello-Handshake:

Once the server received a `Client-Hello-Handshake` message he should reply with this message.

```
0      4      8      40      56      88
+-----+-----+-----+-----+-----+
| Type | Flags | Client ID | Upload-port | Checksum |
+-----+-----+-----+-----+-----+
```

Upload-port [16 Bit]:

The Port where the client should send all following messages to

Flags [4 Bit]:

```
0000
||||
|||+> Client accepted
||+--> Too many clients - connection revoked
|+---> *UNUSED*
+----> *UNUSED*
```

## File-Creation:

Marks the start of a file transfer. Tells the server to create the given file with the given path.  
Replaces existing files.

```
0          4          36          68          72          328
+-----+-----+-----+-----+-----+
| Type | Client ID | Sequence Number | File Type | File SHA3 256 |
+-----+-----+-----+-----+-----+

328          360          424
+-----+-----+-----+
| Checksum | FID Length | FID |
+-----+-----+-----+
```

### File Type [4 Bit]:

```
0000
||||
|||+--> Folder
||+--> Delete folder
|+---> File
+----> Delete file
```

### File SHA3 256 [256 Bit]:

The file SHA3 256 hash to check if the file was transmitted correctly. Unused for folders. [Wiki Link](#)

## File-Transfer:

The actual file transfer message containing the file content.

```
0          4          36          68          72          328          360
+-----+-----+-----+-----+-----+-----+
| Type | Client ID | Sequence Number | Flags | FID SHA3 256 | Checksum |
+-----+-----+-----+-----+-----+-----+

360
+-----+-----+
| Content Length | Content |
+-----+-----+
```

### Flags [4 Bit]:

```
0000
||||
|||+--> First package for the given file
||+--> File content
|+---> *UNUSED*
+----> Last package for the file
```

FID SHA3 256 [256 Bit]:

The SHA3 256 hash of the `FID` to identify which file gets

Content Length [X Bit]: **Size not final. Don't forget about the MTU**

The length of the `Content` field.

Content [defined in "Content Length" in Bit]: **Size not final. Don't forget about the MTU**

The actual file content.

## File-Status:

Used for requesting and responding the current file status e.g. after a connection disconnect.

```
0         4         8                                40         72         134
+-----+-----+-----+-----+-----+-----+-----+
| Type |  Flags | Last Sequence Number | Checksum | FID Length | FID |
+-----+-----+-----+-----+-----+-----+-----+
```

Flags [4 Bit]:

```
0000
||||
| |+-> Request status of FID
| | +--> FID status response
| +---> *UNUSED*
+----> *UNUSED*
```

Last Sequence Number [32 Bit]: The last acknowledged `Sequence Number`. Ignored if `Request status of FID` is set.

## ACK:

For acknowledging `Ping`, `File-Creation` and `File-Transfer` messages.

```
0         4         36                                68         100         104
+-----+-----+-----+-----+-----+-----+-----+
| Type | Client ID | ACK Sequence Number | Checksum | UNUSED |
+-----+-----+-----+-----+-----+-----+-----+
```

ACK Sequence Number [32 Bit]:

The acknowledged `Sequence Number` or `Ping Sequence Number`

## Transfer-Ended:

Gets send by the client once he wants to end the transfer/close the connection.

```
0         4         8         40         72
+-----+-----+-----+-----+
| Type | Flags | Client ID | Checksum |
+-----+-----+-----+-----+
```

Flags [4 Bit]:

```
0000
||||
|||+ -> Transfer finished
||+- -> Cancelled by user
|+--- -> Error
+---- -> *UNUSED*
```

## Ping:

This message is used for ensuring the opponent is still there. The opponent should acknowledge each received `Ping` message with an `Server-ACK`. Should get send by each side if there was no message exchange for more than 5 seconds.

It also can be used for package loss and throughput tests with a modified `Payload Length`.

```
0         4         36         64
+-----+-----+-----+-----+
| Type | Ping Sequence Number | Payload Length | Payload |
+-----+-----+-----+-----+
```

Ping Sequence Number [32 Bit]

An unique number for identifying each ping

Payload Length [28 Bit]:

Describes how long the the following payload is in byte

Payload [X Byte]:

Defined via the `Payload Length`

## Process example:

```
Client                                     Server
| Client-Hello-Handshake                 |
| -----> | The clients starts the connection on the
default port                             |
|                                           | and tells the server the port on which he
listens for answers                      |
| Server-Hello-Handshake                 |
| <----- | The server responds with a client ID and
a port where the                         |
|                                           | server is listening for incoming transfer
messages                                |
| File-Creation                         |
| -----> | The client sends this message to inform
the server about                        |
|                                           | the new file that will be transferred
| Server-ACK                             |
| <----- |
|                                           |
| File-Transfer                         |
| -----> | The client starts sending the file
|                                           |
| File-Transfer                         |
| -----> |
|                                           |
| File-Transfer                         |
| -----> |
|                                           |
| File-Transfer                         |
| -----> |
|                                           |
| Server-ACK                             |
| <----- | The server sends an ACK message for each
message                                |
|                                           | it received from the client
| Server-ACK                             |
| <----- |
|                                           |
| Server-ACK                             |
| <----- |
|                                           |
| Transfer-Ended                       |
| -----> | The client tells the server that the
transfer finished
```