**The file sharing protocol**

The purpose of this document is to detail the protocol through which the clients and the server can transfer information regarding shared files in a safe, reliable, and easy to maintain way.

We’ll first begin the discussion with the following question – what defines a shared file?

Shared files

A shared file can be defined as a unique file which is shared from one or more clients.

The server must contain relevant metadata regarding the shared file and its origin (clients that contain it and willingly share it).

The lifetime of a shared file starts when an end user shares a file which is accessible from the client’s computer.

The file’s metadata is sent to the server, including a unique identifier (in our case – a hash function performed on its data) which is used to distinguish it from other files with otherwise identical metadata.

Note that we have no actual way to distinguish 2 completely identical files (which share the same creation time, data, and name).

Once a client shares a file, the server needs to save its metadata so that it can later respond to clients file searches.

Now that we’ve successfully defined the definition of a shared file – we’ll list the information the client needs to pass to the server in the process of sharing a new file.

Shared file table

* Unique ID (a hash of the file’s data).
* File name – a string containing the name of the file.
* Modification time – the date in which the file was last modified.
* Size – the size of the file, in bytes.

In order to keep track of which client shares which files, we would first need to save information about our clients.

Origins

We’ll now discuss origins – what defines a single origin (or alternatively a single client) and how can we save metadata regarding different origins.

Since clients can connect and disconnect from the server, we would like to devise a protocol which allows us to save information about a single client spanning different connections.

In order to do so we must be able to uniquely identify a client – this will be done via a unique identifier generated once and used across all connections with a single client.

This unique identifier will be used by the server to make sure the client is recognized across multiple connections and its sharing status won’t be lost.

The unique id generation will be done by the server once a client connects, via the following protocol:

Client -> server: ClientID message (“no ID”)

Server -> client: ClientID message (<new unique id>)

From now on new connections from the client to the server in the following way:

Client -> server: ClientID message (<unique id>)

Information about origins is saved in the “origins” table, and holds a single data member in each line – the client’s unique id.

Origins table

* Unique\_id

Connecting a file to its origin

Since the purpose of the files’ metadata is to enable them to be shared across clients, we must save the connection between a shared file and the clients that share it.

This is done via the “shares” table – which holds pairs of unique ids, one from each table (the files table and the origins table).

The server will direct clients to download the file from one or more of its origins in accordance to the current connected sharing clients (if no sharing client is currently connected, the file won’t be downloadable).

Shares table

* File unique id (from the “files” table)
* Origin unique id (from the “origins” table)

The following UML will demonstrate the relationship between the 3 tables mentioned in this document:

