Introduction:

The protocol we’re using is a hybrid between UDP and TCP: even though it requires connection between hosts (in this case, they are Controller-Render and Render-Server) to transfer data, it still can provide a procedure for application programs to send messages to other programs with a minimum of protocol mechanism (just like UDP).

Connection:

Render will provide its ip address and port which will be used Controller and Server to set up a connection between them (similar to UDP handshake).

All connection establishment is based on 2-Way handshake (No ACK). Since all socket is assumed to be steady, 2-Way handshake works. All packets (message) is best-efforts, which did not have lost protection. Furthermore, No timeouts is used more packet lost check.

All different messages is used for a string added header for differentiate different commands.

Commands:

1. Sttn [Filename]:  Once received this by Controller, Render will send a request to Server to stream the file named [Filename]
2. li: Used by Controller, this request will be sent to Render, through that Server with return a list of streamable files.
3. ppp: Controller uses this command to request Render to pause streaming
4. sss: This command can only be used when the streaming session is paused, allows user to use Controller to control Render to resume the stream.
5. Rsrs: This command can only be used when the streaming session is paused, allows user to use Controller to control Render to restart the stream from the beginning.