Client

Developed with Visual Studio 2022.

Client code written with C++.

Boost Library. https://www.boost.org .

Crypto++ Library. https://www.cryptopp.com .

These instructions apply for x86 configuration.

Description of the client operation scheme:

According to me.info file: Sends a registration request, generates RSA pair, and sends public key.Sends reconnect request.After approval connection to server, encrypts file with AES key and attempts to send the encrypted file to server up to 4 times.

Component:

TransferInfo - Reads and analyzes the information from the file transfer.info, the file contains a connection address to the server, a client name and a file path to send to server.

UUID - Converts UUID to string and vice versa.

RSAWrapper - Generates RSA pair, loads the keys. Encrypts and decrypts using the RSA pair.

MeInfo - Reads and analyzes the information from the file me.info, or writes details to it. The file may contains name, UUID and private key.

Packages- Contains structs for requests and responses according to the protocol.

CRC32 - Calculates cksum according to linux cksum.

Base64Convertor - Converts strings to base 64 and vice versa.

Communicator- Used as an interface for communication with the server, sending and receiving packets.

Client - The main class. In which the process of sending requests to the server and receiving responses from it according to the protocol is carried out, using the Communicator interface.It handles the processing of the data, building the requests and deciphering the responses.

In case of an error that cannot be handled by the client, or a server error, an appropriate message will be displayed and the process will be stopped.

Server

Developed with PyCharm 2022.3.1.

Server code written with Python 3.11.

PyCryptodome module https://pycryptodome.readthedocs.io/en/latest/index.html .

Description of the server operation scheme:

Listens to the address 127.0.0.1 on the port according to port.info file, for each client connection, creates a thread that handles requests and responses. The server saves the client details in a database and the files from the clients in local files.

Component:

sizesInfo - Contains data sizes according to the protocol.

Server - Reads port from port.info listens to the address "127.0.0.1" on port from the file, for each new client connection creates ServerThread, a thread to handle the client.

Database - Creates or loads a database that contains details about clients and files. Contains an interface for manipulating the data.

CRC32 - Calculates cksum according to linux cksum.

Packages- Contains classes for requests and responses according to the protocol.

ServerThread - The main class, receives requests from a single client, acts accordingly and sends appropriate responses. Handles saving files to local files. Manipulates data by the database interface.

In case of a server error, or a protocol error, a message will be sent to the client and an appropriate message will be displayed, and the connection to the client will be closed.