ReadMe, mmn15, Yael Shifman

This is a project comprising a server and a client that communicate with each other, allowing clients to securely transfer files from their computers to storage on the server. The server is written in Python, and the client is in CPP.

The software architecture is based on a client-server model. The client initiates contact with the server, exchanges encryption keys, and then transfers the requested file in encrypted communication. The client ensures that the server received the file correctly by comparing checksums on both sides. If not received correctly, the client attempts to resend up to three times. The entire process is done using a binary protocol over TCP, supporting client requests to the server and server responses to the client.

Requests:

Request Structure from Client to Server:

Request	**Field**	**Size**	**Meaning**
Header	Client ID	16 bytes (128 bits)	Unique identifier for each client
	Version	1 byte	Client version number.
	Code	2 bytes	Request code
	Payload Size	4 bytes	Size of the request content
Payload	Payload	Changing	Content varies based on the request

Request Codes:

1025 - Registration:

Field	**Size**	**Meaning**
Name	255 bytes	ASCII string representing the user's name.

1026 - Public Key Transmission:

Field	**Size**	**Meaning**
Name	255 bytes	ASCII string representing the user's name.
Public Key	160 bytes	Client's public key.

1027 - Reconnection (if client was previously registered):

Field	**Size**	**Meaning**
Name	255 bytes	ASCII string representing the user's name.

1028 - File Transmission:

Field	Size	Meaning
Content Size	4 bytes	File size (after encryption)
File Name	255 bytes	Name of the sent file
Message Content	Changing size	File content - Encrypted by a symmetric key.

1029 - CRC Check Passed:

Field	Size	Meaning
Name	255 bytes	Name of the sent file.

1030 - CRC Check Failed, Resend Request (followed by request 1028):

Field	Size	Meaning
Name	255 bytes	Name of the sent file.

1031 - CRC Check Failed for the Fourth Time, Terminate:

Field	Size	Meaning
Name	255 bytes	Name of the sent file.

Responses:

Server Response Structure:

Request	Field	Size	Meaning
Header	Version	1 byte	Server version number.
	Code	2 bytes	Response code
	Payload Size	4 bytes	Size of the response content
Payload	Payload	Changing	Content varies based on the request

Response Codes:

2100 - Registration Successful:

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.

2101 - Registration Failed: (No additional fields)

2102 - Received Public Key, Sending Encrypted AES Key:

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.
Encrypted Symmetric Key	Changing	Encrypted AES key for the client

2103 - File Received Successfully with CRC:

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.
Content Size	4 bytes	Size of the received file (after encryption)
File Name	255 bytes	Name of the received file
Cksum	4 bytes	CRC

2104 - Acknowledges Receipt of Message, Thank You:

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.

2105 - Acknowledges Reconnection Request, Sends Encrypted AES Key (same as code 2102):

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.
Encrypted Symmetric Key	Changing	Encrypted AES key for the client

2106 - Reconnection Request Denied (client not registered or invalid public key):

Field	Size	Meaning
Client ID	16 bytes	Unique identifier for the client.

2107 - General Error in the Server not handled by previous cases (No additional fields).

In addition, the server will keep the following data on sql tables:

Client Table:

Field	Type	Notes
ID	16 bytes (128 bits)	ID (Unique identifier for each client). Index
Name	String (255 characters)	ASCII string representing the username, including null terminator.
PublicKey	String (160 bytes)	Client's public key
LastSeen	Date and Time	Time of the last received request from the client
AESKey	128 bits	AES key sent to the client

Files Table:

Field	Type	Notes
ID	16 bytes (128 bits)	ID (Unique identifier for each client). Index
File Name	String (255 characters)	ASCII string representing the file name as sent by the user, including null terminator.
Path Name	String (160 bytes)	ASCII string representing the relative path and file name as stored on the server, including null terminator.
Verified	Boolean	Boolean indicating whether the checksum was successfully verified against the client