

Certitude

Secure Group Messaging App

Design Document

Version 1

Purpose

Certitude aims to create a no thrills, secure, and easily-disposable text communication channel between a group of people you trust.

Server Experience:

- Launch application
- Select/Create/Delete a **Chat** file
 - A chat is a named file which contains a list of messages, encrypted with a specific password, no data should be in the clear at all
- Password
 - Enter correct password to access a previous **Chat** file
 - Enter new password if creating a new **Chat** file
- Configure Chat
 - Displays list of previous or default configuration settings
 - Port number
 - Number of connections allowed
 - Accept or change them
- Launch Chat
- Exit application

Client Experience:

- Enter configuration
 - Server ip / port to connect to
 - Provide correct password for Chat

- Connect
- Receive list of usernames that have been on the server, and which are online
- Choose a username to join as

An important note on encryption: All saved file data should be encrypted at all times, with no exceptions. Editing and view of data should only be in temporary application memory, and not in storage. If the program were to crash, or to close unexpectedly before re-encrypting a file, it would leave exposed and potentially corrupted chat files.

An important note on Chat file interpretation: If sequences of characters like asterisks are used to identify separate parts of a Chat file, there should be a strong emphasis on user input handling. Designing the server for rogue input could be difficult, telling clients not to send rogue input is easier, since this is a chat built on trust, we can do that. Input won't be processed unless it is legible when decrypted with the secret asymmetric key. Clients could either ban the use of those symbols, or under the hood do some character escaping.

Decrypted text:

NewMessage***John***Hey *** look at cool* symbols***

Deleted symbols.

Or

NewMessage***John***Hey (***) look at cool(*) symbols(***)

() Gets removed in presentation, but kept during processing

The processor would know not to view asterisks in () as file info, but chat info

Another method could be to encapsulate an entire new message with a specific set of characters, however this wouldn't prevent users from escaping that, it's a lot like an sql injection problem.

Ex:

ServerName //A section title to double check where you are

*** //Three *s indicating section change

Config

* //One * indicating new element in a section (new var, new message, etc.)

Port = 3333

*

Var1 = 123

*

VarJohn = Doe

*

VarColor = Red

Chat

*

Tom:How are you guys?

*

David:I'm doing great how are yo?

*

David:you* //Interpreted as two message ends, results in a blank message

*

Tom:I'm doing just finw!

*

Tom:FINE*** //Three might end chat, or just a bunch of blank messages

*

Systems:

System Interaction: