Secure Fast Chat: Client Program

Khushang Singla \and Mridul Agarwal \and Arhaan Ahmad

Nov 16, 2022

CONTENTS:

CHAPTER

ONE

CLIENT_SIDE

1.1 Message module

class Message.**Message**(*conn_socket*, *task*, *request*)

Bases: object

This is the class to handle Encryption of messages. The format in which the message is sent to server is determined in this class

Parameters

- **task** (*str*) – Task to be done. It can have the values signup, login, send_message
- **socket** (*socket.socket*) – The socket used for connection with Server
- **request_content** (*dict*) – Content to include in the request to send to server
- **_data_to_send** (*bytes*) – Contains the data to send to the server
- **_recvd_msg** (*bytes*) – Content recieved from server is stored here

Constructor Object

Parameters

- **conn_socket** (*socket.socket*) – Socket which has a connection with server
- **task** (*str*) – Task to do. It can have values: login, signup, send_message
- **request** (*str*) – Content to send to server

_send_data_to_server()

Function to send the string to the server. It sends content of _send_data_to_server to the server

_recv_data_from_server(*size*, *authenticated=True*)

Function to recv data from server. Stores the bytes recieved in a variable named _recvd_msg.

Parameters **size** (*int*) – Length of content to recieve from server

_json_encode(*obj*, *encoding*)

Function to encode dictionary to bytes object

Parameters

- **obj** (*dict*) – dictionary to encode
- **encoding** (*str*) – Encoding to use

Returns Encoded obj

Return type bytes

_json_decode(*obj*, *encoding*)
Function to decode bytes object to dictionary

Parameters

- **obj** (*bytes*) – Encoded json data
- **encoding** (*str*) – Encoding used

Returns Decoded json object

Return type json

_hash_password(*passwd*)
Function to salt and hash the password before sending to server

Parameters **passwd** (*str*) – Password to be hashed

Returns Transformed Password

Return type string

_encode(*text*, *key*)
Function to encode the text using key from server

Parameters

- **text** (*str*) – string to encode
- **key** (*str*) – key for encryption

Returns Encoded text

Return type str

_encrypt(*msg*, *key*)
Encrypt the message to send to reciever

Parameters

- **msg** (*str*) – Message to encrypt
- **key** (*str*) – Key to encrypt the message

_create_loginpass_request()
The jsonheader has the following keys: | byteorder, request, content-length, content-encoding.

Returns Message to send to server directly for login

Return type bytes

_create_loginuid_request()
The jsonheader has the following keys: | byteorder, request, content-length, content-encoding. The value for request is 'loginuid' | The content has user id.

Returns Message to send to server directly for login

Return type bytes

_create_signuppass_request()
The jsonheader has the following keys: | byteorder, request, content-length, content-encoding. The value for request is 'signuppass' | The content has encoded password

Returns Message to send to server directly for login

Return type bytes

_create_signupuid_request()
The jsonheader has the following keys: | byteorder, request, content-length, content-encoding. The value for request is 'signupuid' | The content has user id.

Returns Message to send to server directly for login

Return type bytes

_login()
Function to help login into the system. This function sends the login details to the server | The function expects to recieve a response of size 2 from server which gives 0 if success and 1 if wrong uid and 2 for wrong passwd

Returns Response from server converted to int

Return type int

_signuppass()
Function to save account password at server The function expects to recieve a response of size 2 from server which gives 0 if username already taken and 1 for success

Returns Response from server converted to int

Return type int

_signupuid()
Function to help signup to make new account. This function sends the new user userid to the server | The function expects to recieve a response of size 2 from server which gives 0 if username already taken and 1 if username is available

Returns Response from server converted to int

Return type int

_keyex()

_recvmsg()
Recieves the information from server. It interprets this as a message from some user and returns the message recieved. The header of recieved request should at least have 'content','content-type','sender','content-len','byteorder' as the keys

_sendmsg()
This function sends the message to the server

_create_group_key()
Function to get the Private key of group to use it to encrypt the messages being sent in groups

Returns private key

Return type str

_create_grp()
Function to send a request to create a group

processTask()
Processes the task to do

Returns Returns int to represent result of the process. The details of return values are given in the corresponding functions handling the actions.

Return type int

1.2 app module

app.**connectToServer**(*sock*)
Function to connect to server and exchange the key for encryption

Parameters **sock** (*socket.socket*) – Socket variable to use for connection

app.**getUserSecretFromPassword**(*passwd*)
Function to convert password to 'User Secret'

Parameters **passwd** (*str*) – Password to convert

app.**login**(*sock=None*)
Function to help user log in to the app

Returns Socket with which user is connected to server

Return type socket.socket

app.**signup**(*sock=None*)
Function to help user make new account

Returns Socket with which user is connected to server

Return type socket.socket

app.**handleMessageFromServer**(*socket*)
This function is called when there is a message from server. . . .

CHAPTER

TWO

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

INDEX

Symbols

A

C

G

H

L

M

P

S