Network programming – Client/Server message relay over TCP protocol

Implementation of new Security measures for client server message relay

Due to lack of authentication, when users joined the server, any name was able to send messages, now we have each user and their MD5 hashed password on file.

An MD5 hash password cannot be decrypted/converted back to friendly ASCII so we are using MD5 for password checking.

We are also using MD5 for message handling.

Each message transmitted from client to server and vice versa will contain an MD5 hash attached at the head which is determined by a secret code in digits that the server will have and each authorized client should have.

The server initializes with the secret code and each client connects with the secret code.

Any secret code not transmitted correctly will not allow messages to be sent back and forth, this was implemented so that to avoid a brute force attack if we simply rejected the connection upon connection attempt – similar to reaching a website and having access to the server prior to authentication to access private information.

An additional security protocol is to apply a encryption/decryption algorithm to each message transmitted.

We have decided on Fernet, we would have liked to use MD5, but since MD5 cannot reverse encrypt (decrypt) we cannot use this method.

We need a method where we can encrypt and decrypt.

A Fernet key is generated at each instance launch of the server and provides the key.

Each client, needs to know the key and copy/paste this key into each connection attempt.

Successful implementations:

- We have successfully implemented an MD5 hash generation from a given 4 digit integer as a secret code that the server has and each client should have.
- Observed successful transmission of MD5 hashed secret code attached to the head of each message transmission.
- We have successfully implemented an MD5 hash generation for user passwords at client end, and a successful match upon login request for server.
  Passwords are transmitted in md5 hash.
- We have successfully implemented a Fernet encryption/decryption algorithm for all messages received and sent via client and server.
  This encryption covers the entire message sent by client or server.
  The decryption also covers the entire message received by client or server.

Function code:

Passes the secret code and the hash value of it and checks to ensure matches, returns boolean

@staticmethod
  def verify_md5_hash(number, hash_value):
  return hashlib.md5(str(number).encode()).hexdigest() == hash_value

Passes the secret code and generates an md5 hash and returns it's value.

@staticmethod
  def generate_md5_hash(number):
  return hashlib.md5(str(number).encode()).hexdigest()

Blockers and known issues:

- If in the event the server does not receive the fernet encryption from the client side, it raises an InvalidToken error message and closes the program – need to work on avoiding this.
- At this time the function to offer a read receipt has not been implemented.
- We attempted to take the feedback from the previous version to improve it, but ran into a lot of message transmission issues along with common OOP issues with passing variables to functions that also needed to call (self) functions, so we kept to the same version and added the security functionality required.