**CY2004**

**Cyber Security**

**Assignment 02**

**Secure Client Server Model Testing**

**Submitted by:** **Saad Bukhari**

**Roll number:** **23I-2001**

**Date:** October 13, 2024

**Table of Contents**

[**Introduction** 3](#_Toc179715837)

[**Testing:** 3](#_Toc179715838)

[**Key Exchange** 3](#_Toc179715839)

[**Registration** 3](#_Toc179715840)

[**Login** 5](#_Toc179715841)

[**Encryption** 5](#_Toc179715842)

[**Logging** 6](#_Toc179715843)

[**Conclusion:** 7](#_Toc179715844)

**Secure Client Server Model Testing**

# **Introduction**

The client-server model is a fundamental architecture in network computing, enabling communication and data exchange between two entities: the client and the server. In this assignment, the client-server model is tested using Python's socket programming, implementing secure communication through Diffie-Hellman key exchange and AES encryption. The system ensures that user credentials are handled securely during registration and login processes, with an emphasis on password strength, validation, and encryption. This report will document the testing and analysis of the system’s features, including user registration, login, and encrypted messaging.

# **Testing:**

## **Key Exchange**

The first process of the client server model is the key exchange. Here we have established a secure functionality of key exchange for AES encryption. This encryption will encrypt the credentials sent to server during the registration and login phase.

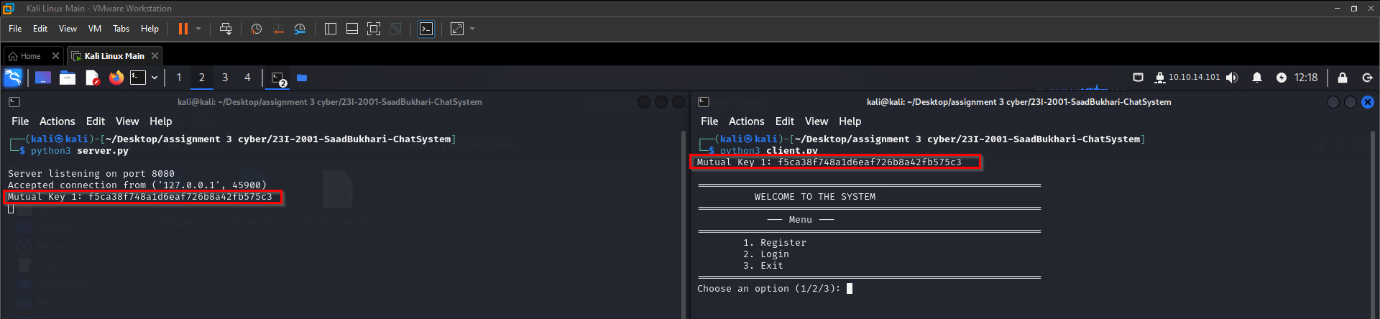


Figure 1: Mutual Key 1

Here we can see as soon as clients connect to the server, on both side the mutual key is established which will encrypt the Credentials during the Registration and Login Phase.

## **Registration**

So, first of all We Have A Registration Phase. In this phase the user will send their credentials to the server for registration. Here to ensure the transfer of credentials we will be using Mutual Key 1 for which was established using Deffie-Helman.

Let’s send the credentials during registration and check if they are exposed or not.

I have also implemented check for correct email, username and password can’t be same and password security. To ensure confidentiality I have introduced a feature which would not show password while entering.

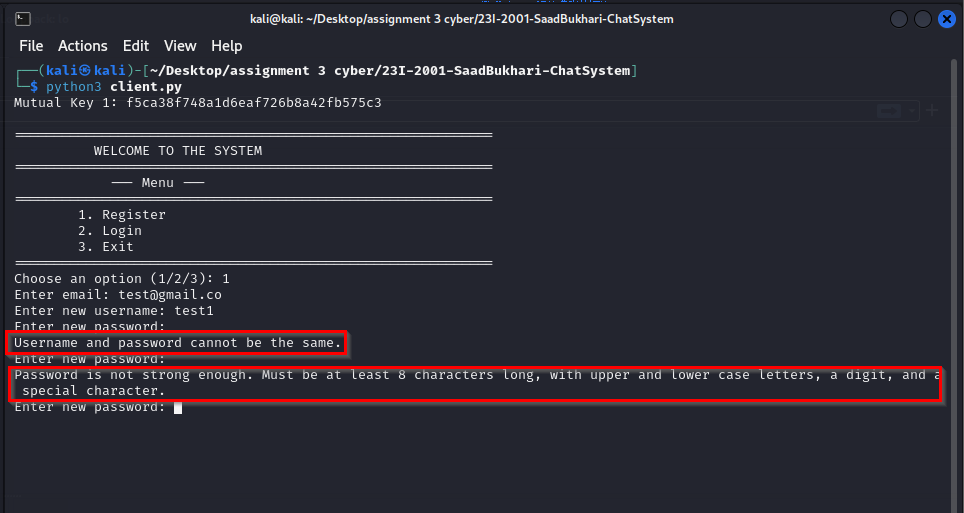


Figure 2: User Validation

Now sending the credentials Let’s verify the registration

A screenshot of a computer

Description automatically generated

**Figure 3: Secure Credentials**

Here we can see the credentials transmitted through the channel are secured.

Now let’s check how credentials are stored in the database.

In database, we have email, username and password stored. Password is stored in sha-256 hashed with a 16 bit random salt which makes it hard to crack password using dictionaries.

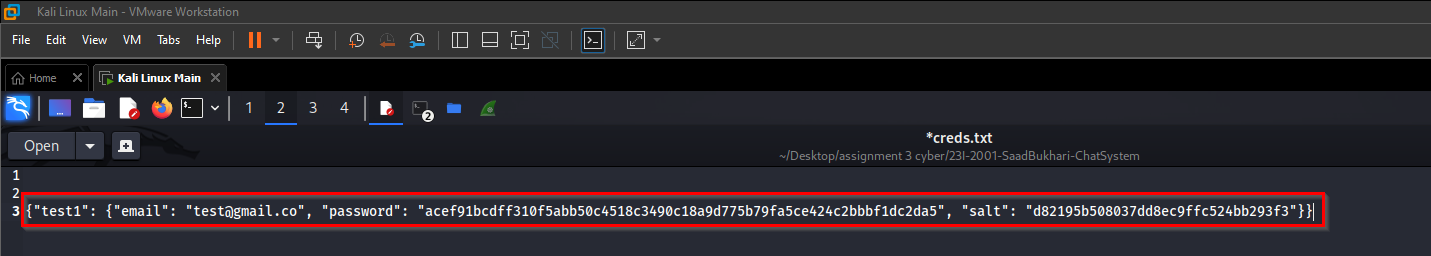


Figure 4: Secure Database

We can verify that credentials are stored in a safe way in database.

## **Login**

Now for login, it will use the same key calculated at start.

So Let’s verify the Login Validation and login with the user we have created

A screenshot of a computer

Description automatically generated

Figure 5: Login Validation

Here we can see we cannot login with any other password and password is also hidden while entering. Same validation for the user. If user not found it gives error.

In case of correct verification, it logs in the users.

Now let’s verify the secure credentials transmission to the server

A screen shot of a computer

Description automatically generated

Figure 6: Secure Transmission

Here we can see out credentials are transferred securely. All messages are transmitted in a secure way.

## **Encryption**

Now for chatting phase, First a new mutual key would be created by using username+(oldkey) and then it will be used to encrypt all the messages.

We can see the creation of new key

A screenshot of a computer

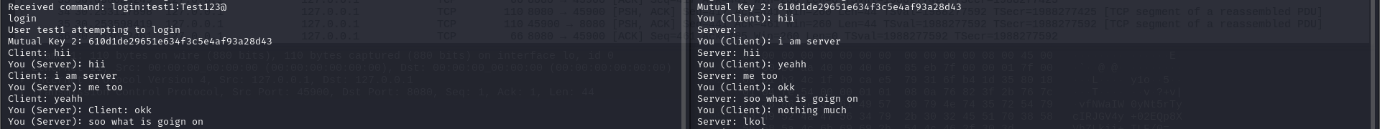
Description automatically generated

On sever side

A white text on a black background

Description automatically generated

Now let’s verify secure chatting system



Now In Wireshark

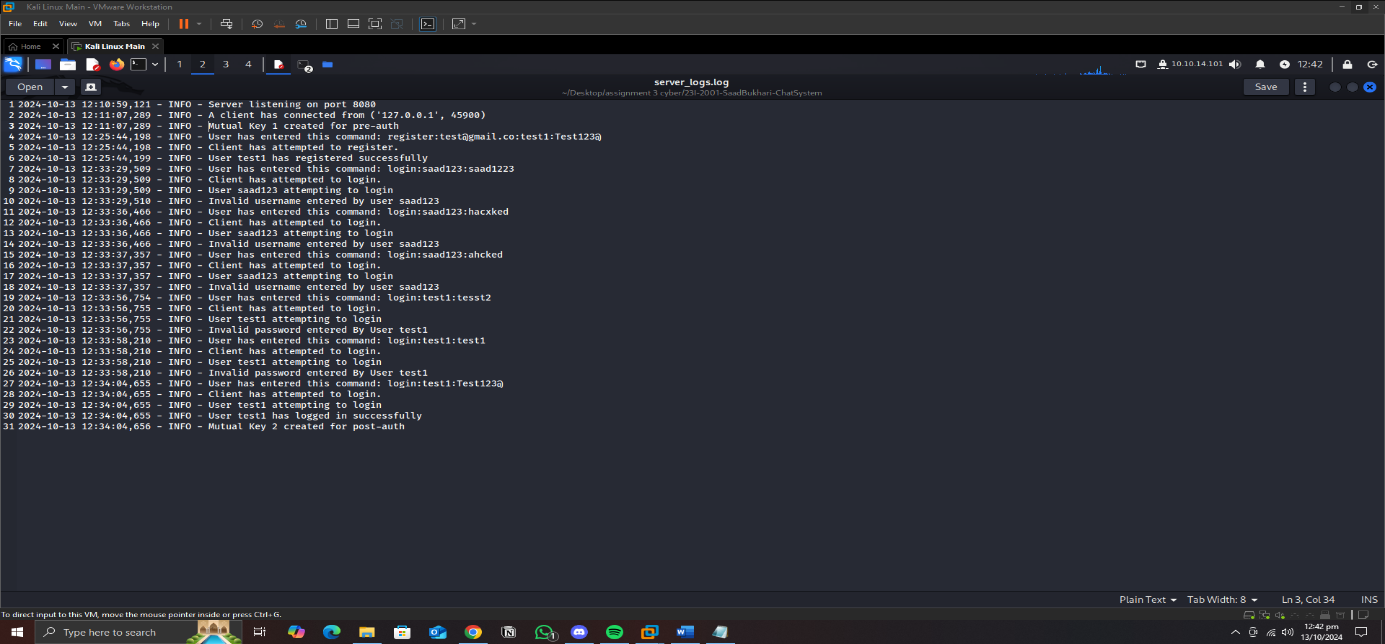
A screen shot of a computer

Description automatically generated

So It is clear that our chatting is also secured using AES

## **Logging**

This was not required but added as a bonus feature which would log all the important communication and actions performed by user. It logs all the things and saves them in a log file



# **Conclusion:**

The testing of the client-server model demonstrated a secure and efficient communication mechanism between the client and server. The integration of Diffie-Hellman key exchange and AES encryption successfully safeguarded data during transmission. Through rigorous validation of user credentials and password strength checks, the system prevented common security vulnerabilities. The encrypted messaging was reliable, and the mutual keys provided an additional layer of security. Overall, the assignment fulfilled its objective of creating a robust, secure client-server system, ensuring data confidentiality and integrity.