**Secure Chat System with Registration, Login, and**

**Encrypted Communication Test Report**

Contents

[Registration Testing 1](#_Toc179407269)

[Login Testing 3](#_Toc179407270)

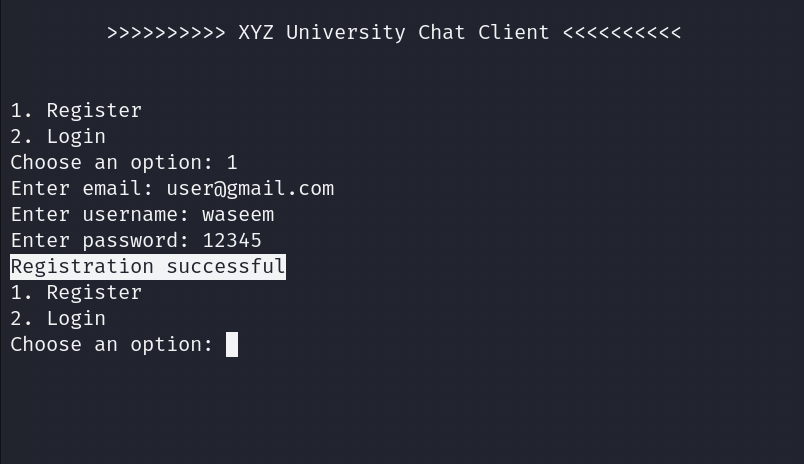
[Security testing 5](#_Toc179407271)

[Encryption testing 7](#_Toc179407272)

# Registration Testing

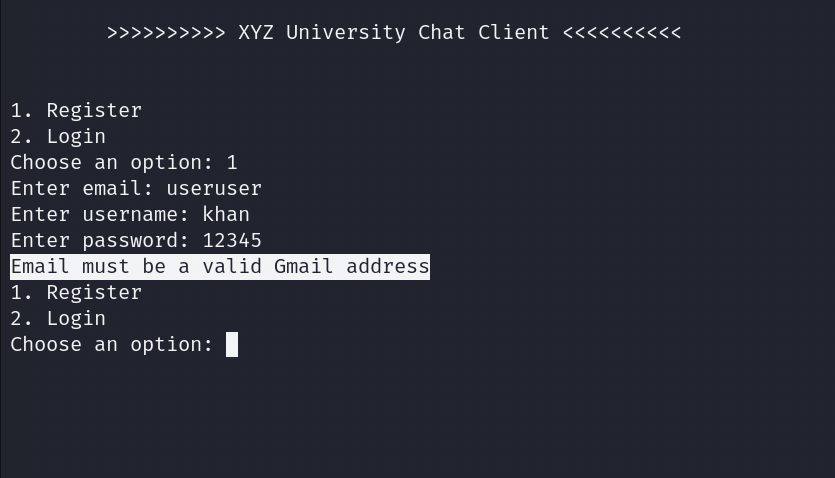
**Test Case 1: Valid Registration**

* **Input:**  
  Email: user1@gmail.com  
  Username: waseem  
  Password: 12345
* **Expected Result:**  
  The server should accept the registration and return a success message: "Registration successful".
* **Actual Result:**  
  The server returned: "Registration successful".



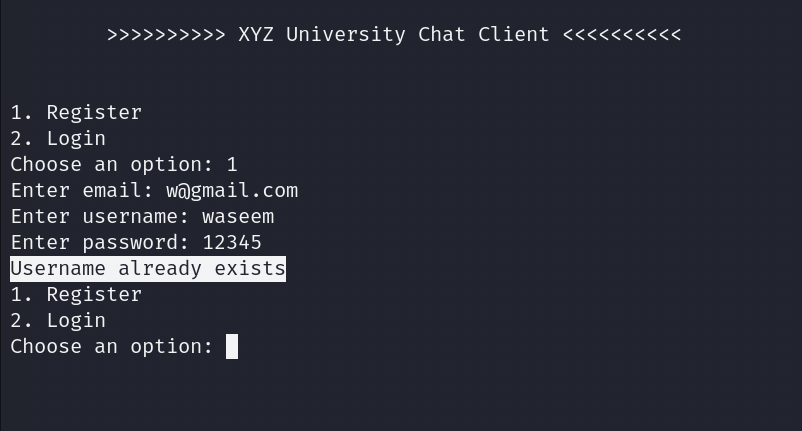
**Test Case 2: Invalid Email Format**

* **Input:**  
  Email: useruser  
  Username: khan  
  Password: 12345
* **Expected Result:**  
  The server should reject the registration and return an error: "Email must be a valid email address”.
* **Actual Result:**  
  The server returned: "Invalid email. Only @gmail.com addresses are allowed."



**Test Case 3: Duplicate Username**

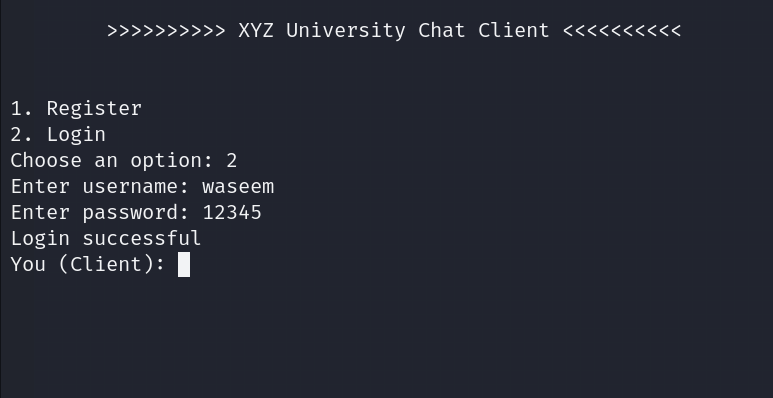
* **Input:**  
  Email: w@gmail.com  
  Username: waseem  
  Password: 12345
* **Expected Result:**  
  The server should reject the registration and return an error: "Username already exists".
* **Actual Result:**  
  The server returned: "Username already exists".



# Login Testing

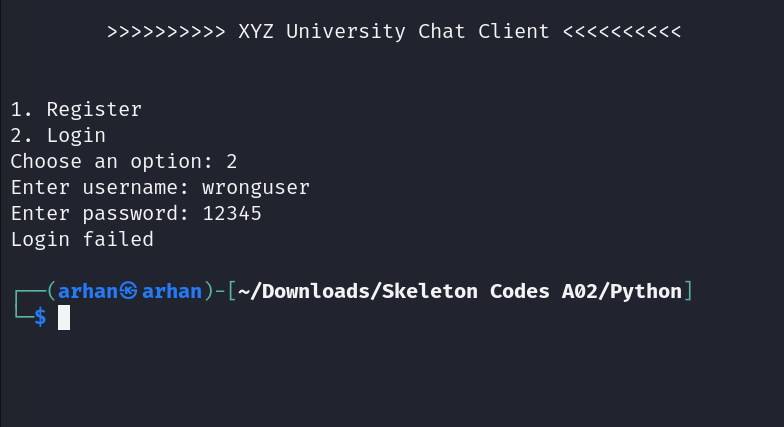
**Test Case 1: Successful Login with Valid Credentials**

* **Input:**  
  Username: waseem  
  Password: 12345
* **Expected Result:**  
  The server should authenticate the user and return a success message: "Login successful".
* **Actual Result:**  
  The server returned: "Login successful".



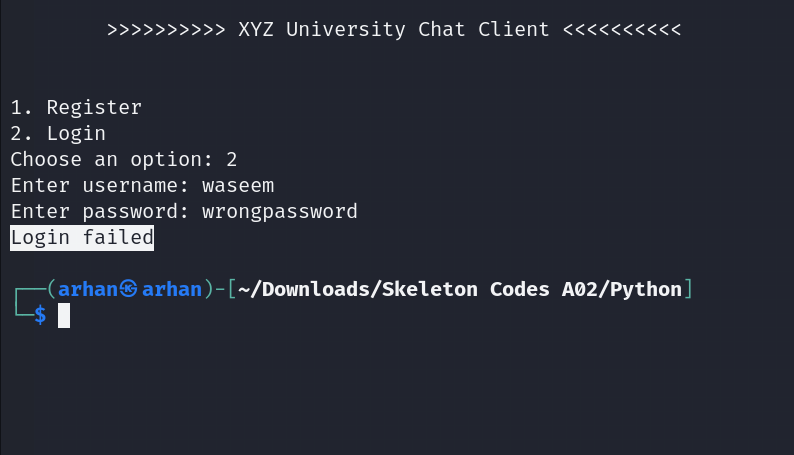
**Test Case 2: Failed Login with Incorrect Username**

* **Input:**  
  Username: wronguser  
  Password: 12345
* **Expected Result:**  
  The server should reject the login attempt and return an error: "Login failed".
* **Actual Result:**  
  The server returned: "Login failed".



**Test Case 3: Failed Login with Incorrect Password**

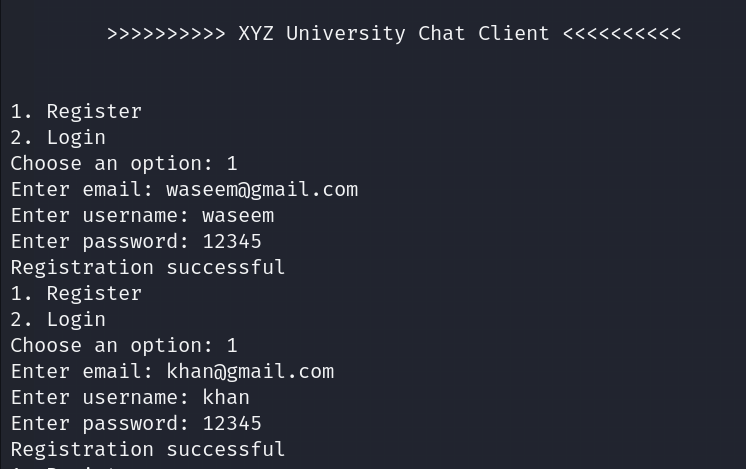
* **Input:**  
  Username: waseem  
  Password: wrongpassword
* **Expected Result:**  
  The server should reject the login attempt and return an error: "Login failed".
* **Actual Result:**  
  The server returned: "Login failed".



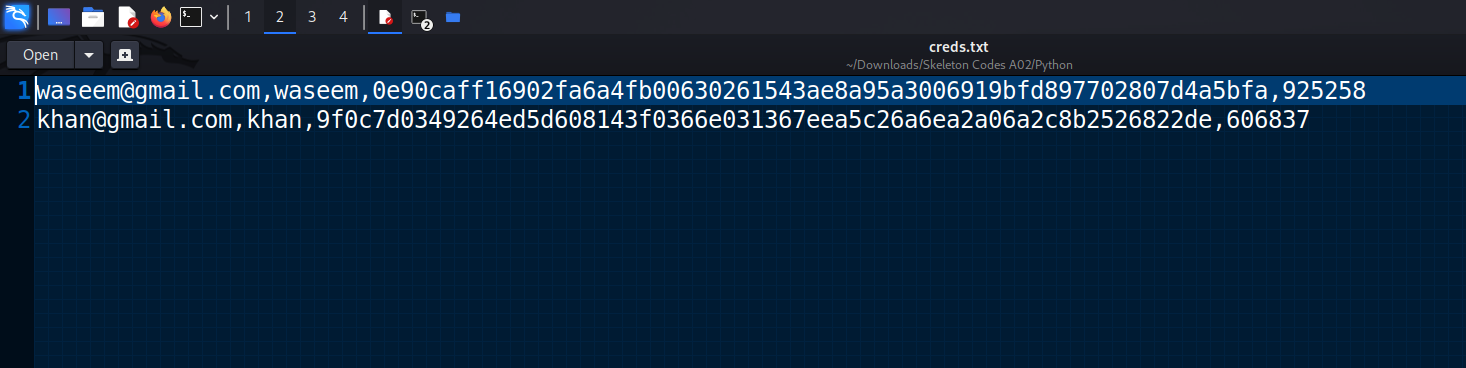
# Security testing

**Test Case 1: Identical Passwords Generate Different Hashes (Salting)**

* **Input:**
  + Register two different users (waseem and Khan) with the same password: 12345.
* **Expected Result:**  
  Thanks to random salting, the server should generate different password hashes for waseem and Khan despite using the same password.
* **Actual Result:**  
  Hashes for user1 and user2 differed, confirming that the salting mechanism works correctly.



**Result from cred.txt file.**

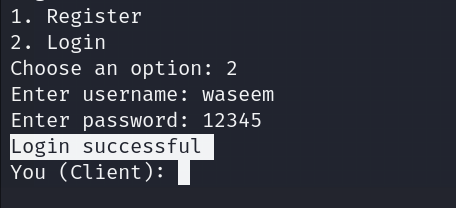


This file demonstrates that password hashes are stored securely using SHA-256 hashing along with random salts, ensuring that identical passwords generate different hashes.

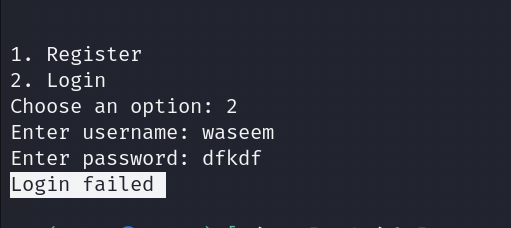
**Test Case 2: Verifying Password During Login**

* **Input:**
  + Login with the correct username (waseem) and password (12345).
  + Login with the correct username but incorrect password.
* **Expected Result:**
  + The server should authenticate the user successfully when the correct password is provided.
  + The server should reject the login attempt when the password is incorrect.
* **Actual Result:**
  + Login succeeded when using the correct password.
  + Login failed with the incorrect password.

**Correct username and password.**



In this screenshot, a successful login attempt is made with the username "waseem" and the correct password "12345". The system confirms the login with the message "Login successful", indicating thatthe login verification process is functioning correctly for valid credentials. Correct username but incorrect password.



In this screenshot, a login attempt is made with the username "waseem" and an incorrect password "dfkdf". The system correctly responds with "Login failed", indicating that the password verification process is functioning as expected.

This test case demonstrates that the login system is secure and rejects incorrect credentials.

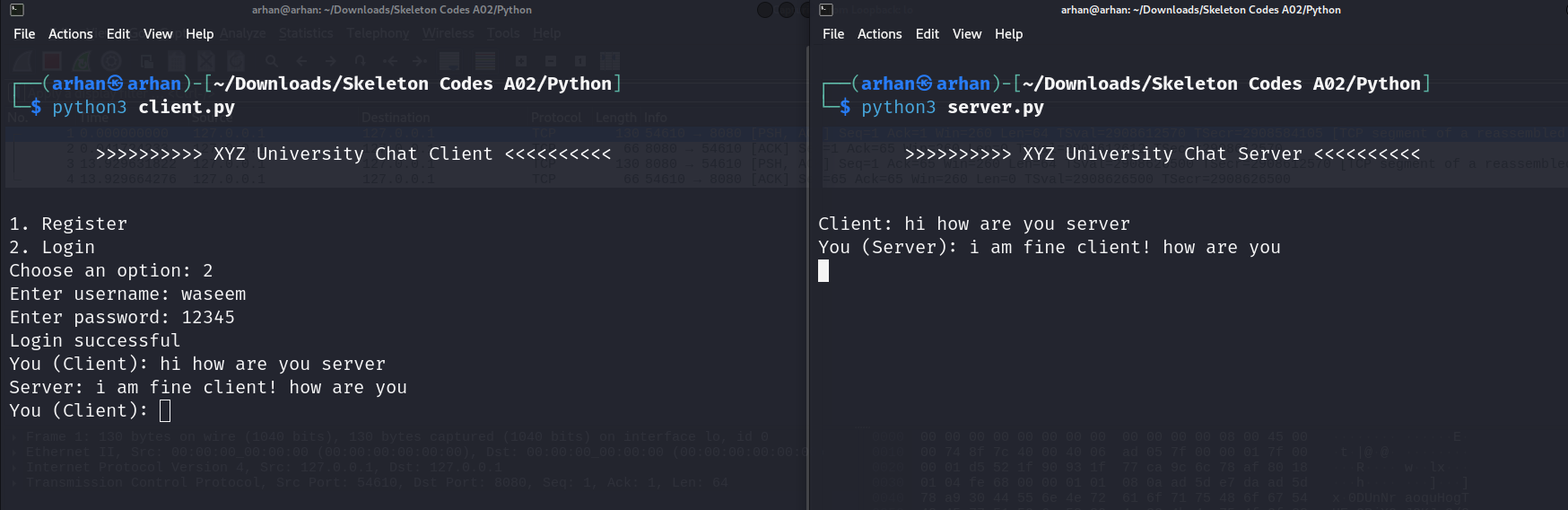
# Encryption testing

**Test Case:**  
Verify that communication between the client and server is properly encrypted during and after login and that decryption works correctly at both ends.

* **Steps:**
  1. Capture network traffic using Wireshark while logging into the client-server chat system.
  2. After login, exchange several messages between the client and server to test encryption and decryption.
  3. Inspect the captured packets in Wireshark to ensure the communication is encrypted and no sensitive information is visible in plaintext.
  4. Verify proper decryption of the messages on both client and server sides.
* **Expected Result:** After login, all communication should be encrypted during transit, and both the client and server should be able to decrypt the messages correctly. When inspecting the packets with Wireshark, the content should be in cipher text, ensuring confidentiality.
* **Actual Result:** After logging in and exchanging messages:
  1. Both the client and server successfully decrypted the messages, confirming the proper use of AES-128 CBC decryption.
  2. When reviewing the packet capture in Wireshark, all the captured data appeared encrypted and in cipher text, making it unreadable to anyone monitoring the network traffic.
  3. The captured packets did not show any sensitive information like plain text messages, usernames, or passwords, ensuring that the communication between the client and server maintained confidentiality.

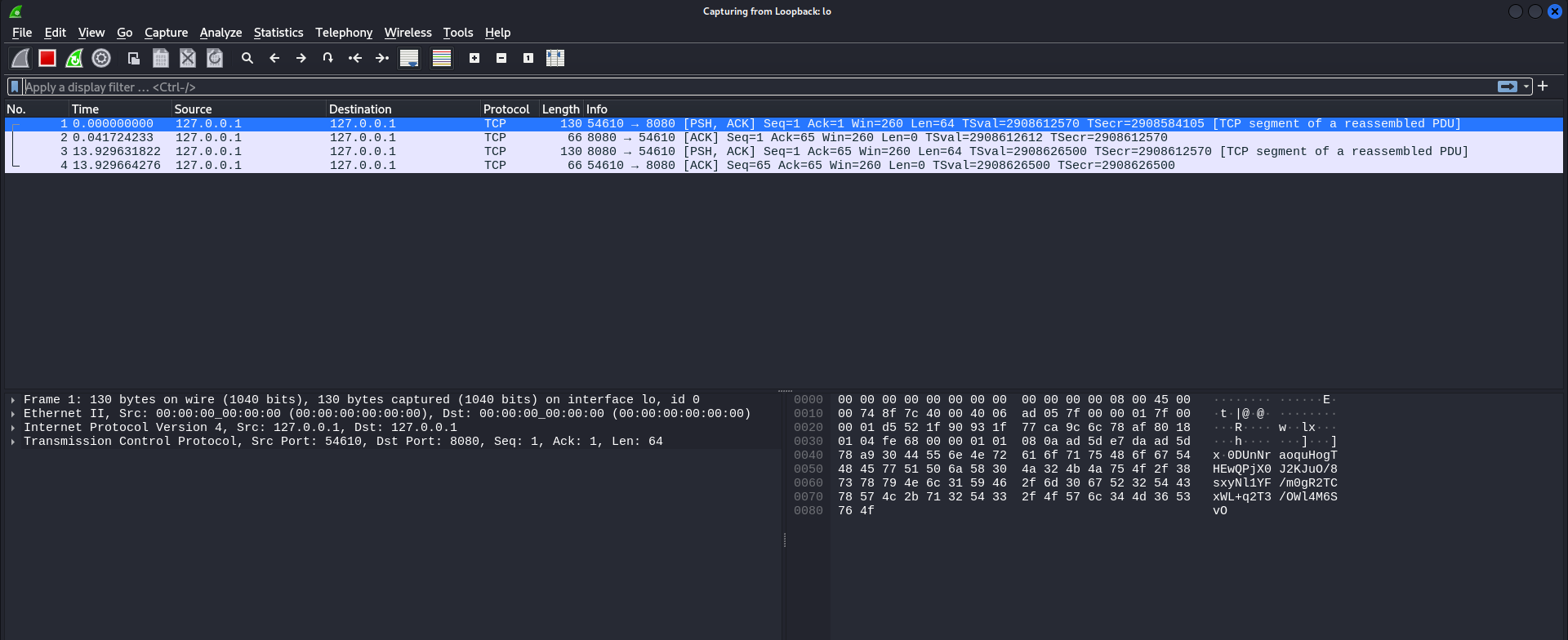
**Confidentiality Assurance:** The encrypted messages in the captured packets demonstrate that the system upholds confidentiality by ensuring that no unauthorized party can read the communication between the client and the server. This is critical for securing user information and private conversations.

**Evidence:**

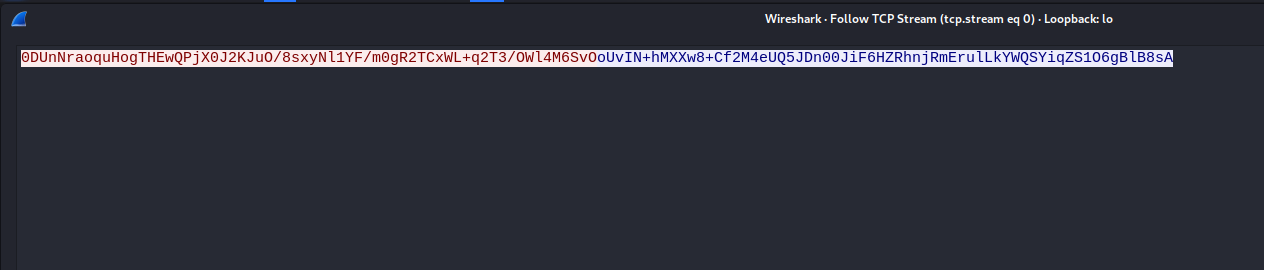


Some messages are exchanged between server and client and both are decrypted at both ends properly, ensuring a proper decryption mechanism.

Now with the help of wireshark, we will sniff some packets and check what is in the communication.



As we can see some packets are successfully sniffed, now we will see the content of the packet, let’s see what we have captured.



The text is successfully encrypted ensuring proper encryption mechanism and the text is meaningless ensuring confidentiality. If someone captures the communication packets, he or she will not understand what is shared between the server and clients.