**Assignment 2 Report – Encrypted Chat Application**

**Workflow**

1. Client generates Public Key and Private Key using RSA Algorithm

2. Client generates Message Digest (Hash value) of Public Key

3. Client sends Public Key to Server

4. Once Server receives the Public Key from the Client, it generates its Hash value and sends "YES" acknowledgement to Client

5. Client then sends Hash value of its Public Key to the Server

6. Server then compares the Hash received with its own calculated Hash value of Client's Public Key

7. Once the Client is authenticated based on the credibility of the public key, the Server then proceeds with generation of Session Key

8. Server calculates the Hash value of this Session key and then encrypts the Session Key using the Client's Public Key and sends the encrypted Session Key to the Client

9. Client will use its Private Key to decrypt the encrypted message received from Server to get the original Session Key

10. Client then generates the Hash value of this Session Key for further communication with the Server

11. At this point, handshake is complete between the Client and the Server and then the actual transmission of messages begins

12. Client initiates the transmission of messages by accepting Input from User

13. Client first encrypts the Input with the Hash value of Session Key using AES algorithm and then sends it to the Server

14. The Server uses the same hash value of Session Key to decrypt the message received from Client

15. The Server then responds with another message following the same encryption process using Session Key

16. The Client finally decrypts the received encrypted message and this process continues until either of them sends "exit"

**Constraints:**

1. The input should be given in string format from the console

2. The "exit" message should be entered in lowercase to terminate the session

**CIA has been achieved in the following ways:**

**Authentication**

Basically, the Server needs to know the authenticity of the Client before a Session can be initiated. So, this has been achieved by using RSA module wherein the Client generates its Public and Private Keys and then sends the Public Key to the Server. The server when in future encrypts the Session Key using the Client's Public Key then, only the client can decrypt it using its Private Key. So later, when Client uses the Session Key to encrypt a message and the Server uses the same Session Key to decrypt the message, it simply explains the authenticity of the Client.

This is only possible if the owner of the Private Key could decrypt the encrypted Session Key sent by the Server.

**Integrity**

Hashing Algorithm (SHA-1) is used to achieve integrity. Firstly, the Client sends the original Public Key to the Server and once the Server receives the Public Key, it then calculates the hash value using Hashlib module. The client then sends the Hash value of its Public Key to the Server and the server compares its Hash value with the received one.

If it matches, it signifies that the data hasn't been corrupted and has maintained its integrity.

**Confidentiality**

The server and client maintain the confidentiality by using Session Key to encrypt the messages that is being sent to and fro.

**Packages Used –**

* Crypto – Cipher, AES, PublicKey
* Hashlib
* Socket
* OS
* Time
* Sys