(1)

MeetingRoomClient.java – uses private key to encrypt session key

MeetingRoomServer.java – uses public key to decrypt session key

LoginCreds.txt – Server checks this file for authorized logins

MeetingTimes.txt – Server checks this file for timeslots

PublicKey.txt – generated from Lab 3 program, 2048 bit key

PrivateKey.txt – generated from Lab 3 program, 2048 bit key

(2)

a. Protocol Message Format

Client generates a 128 bit session key

Client uses private key to encrypt the session key

Client sends “You are ready to reserve the room” special acknowledgement message to server

Client sends OAEP encrypted session key to server

Server receives special acknowledgement message

Server decrypts the encrypted session key using the public key

Server encrypts special acknowledgment message using session key

Server sends encrypted special acknowledgement key to client

Client receives encrypted special acknowledgement

Client decrypts special acknowledgement message using session key

Client matches the created special acknowledgement message to the message received from server

If match, then the key exchange is successful, and client is prompted to login

Client is able to log in and communicate with the server to reserve timeslots

If there is no match, acknowledgement fails and the program exits

b.

Successful key exchange screenshots:

Client View:

Graphical user interface, text, application

Description automatically generatedServer View:

Graphical user interface, text, application

Description automatically generated

The key exchange was successful and the client program is able to login.

Successful login and reserve Client View:

Graphical user interface, text, application

Description automatically generated

Successful login and reserve Server View:

Graphical user interface, text, application

Description automatically generated

The program now works with the client using their private key to encrypt the session key, the server decrypts the session key using the public key, then the key exchange using the special acknowledgment and session key as encryption/decryption is verified before the client and server continue on with login and reservation functions.

The only part that I could not get to work was using the OAEP padding type, I was running to errors trying to encrypt the session key using OAEP padding. The program uses PKCS1 padding to encrypt and decrypt the session key as I was able to get that padding type to function properly.