1.

MeetingRoomClientLab6.java – contains the digital certificate exchange and the 4 step key exchange protocol for establishing a session key based on Fig 15.5

MeetingRoomServerLab6.java - contains the digital certificate exchange and the 4 step key exchange protocol for establishing a session key based on Fig 15.5

Certificate\_Authority.java – used to generate the digital certifications with the CA

**Files used:**

clientKeyStore.jks – clients digital certification from Certificate\_Authority.java program

ClientPrivateKey.txt – client’s private key, 1024 bit

ClientPublicKey.txt – client’s public key, 1024 bit

keystoreCA.jks – CA keystore file for verification

LoginCreds.txt – authorized login combinations for reservations

MeetingTimes.txt – All available and reserved meeting times for reservations

receivedclientcert.jks – Server creates and uses this file sent from client to verify digital certificate

receivedservercert.jks – Client creates and uses this file sent from server to verify digital certificate

serverKeyStore.jks – server’s digital certification from Certificate\_Authority.java program

ServerPrivateKey.txt - server’s private key, 1024 bit

ServerPublicKey.txt – server’s public key, 1024 bit

Key Exchange Protocol:  
Step 1: E(PUb, [N1 || IDA])

Client(A) sends an encrypted message containing a random nonce and the IDA which is any identifying factor to Server(B). Encrypted using PUb which is B’s public key, B decrypts the message using PRb(B’s private key) to see N1

Step 2: E(PUa, [N1 || N2])

Server(B) will generate a random nonce N2. Then send a message containing N1 and N2 back to Client(A). It will be encrypted using PUa, A’s public key. A will decrypt it using PRa, A’s private key and check to see if B’s N1 reply matches the N1 sent to defend against a replay attack.

Step 3: E(PUb, N2)

Client(A) sends an encrypted message containing N2 to Server(B), encrypted using PUb, B’s public key. B decrypts the message using PRb, B’s private key and checks to see if N2 received matches the N2 sent. This ensures to defend against a replay attack.

Step 4: E(PUb, E(PRa, Ks))

Client(A) encrypts the session key(Ks) using their private key(PRa). Then A encrypts the encrypted session key using Server(B)’s public key(PUb) and sends it to the server. B then decrypts the message using their private key(PRb), and then decrypts it again using A’s public key(PUa). A and B now have established a session key that can be used for further encryption and decryption of the reservation program. Key exchange protocol is complete at this point.

Screenshots of process:

Server waiting for client connection to begin:

Graphical user interface, text, application

Description automatically generated

Client view when connecting successfully:

Graphical user interface, text, application

Description automatically generated

Digital certificate is received from server, verified, and then the key exchange protocol begins and progresses through each step. Once it is complete and the session key is established then the client can login.

Server view:

Text

Description automatically generated

Digital certificate is received from client, verified, and then the key exchange protocol begins and progresses through each step. Once it is complete and the session key is established then the client can login and server waits to check the login credentials.

Client Fails 5 times, disconnected due to too many failed attempts:  
Client view:

Text, email

Description automatically generated

Server view:

Graphical user interface, text, application

Description automatically generated with medium confidence

Client successfully logs in and reserves a timeslot:  
Client view:

Graphical user interface, application

Description automatically generated

Server view:

Graphical user interface, text

Description automatically generated

If client chooses to not reserve again program exits and server restarts:

Client view:

Graphical user interface, text, application, email

Description automatically generated

Server view:

Graphical user interface, text, email

Description automatically generated

If client says y, then they are prompted to login and make another reservation:

Client view:

Graphical user interface, text, email

Description automatically generated

Server view:

Graphical user interface, text, application, email

Description automatically generated

If all time slots are reserved, client is notified and program exits:  
Client view:

Graphical user interface, text, application, email

Description automatically generated

Server view:

Graphical user interface, text

Description automatically generated