Lab 6
Gunnar Yonker

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.

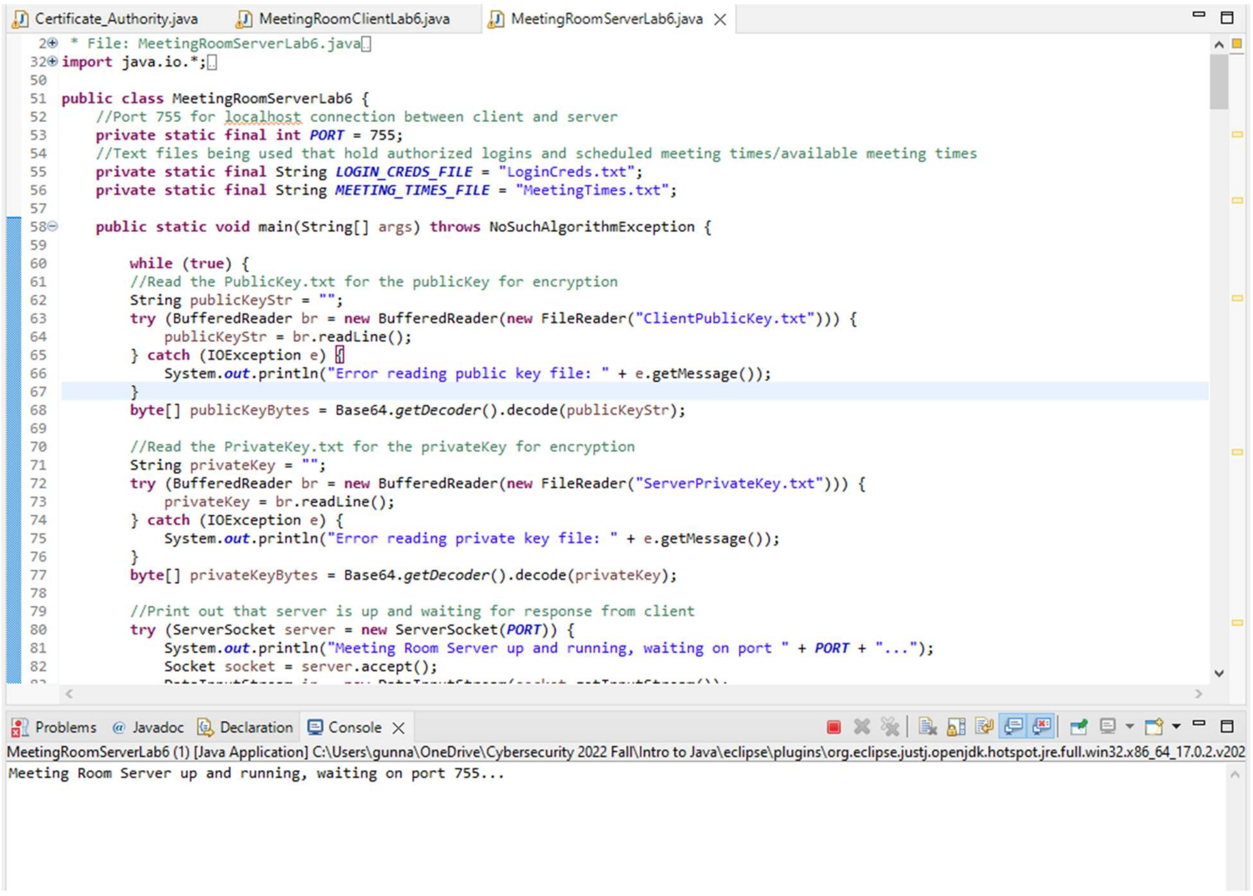
Lab 6
Gunnar Yonker

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:

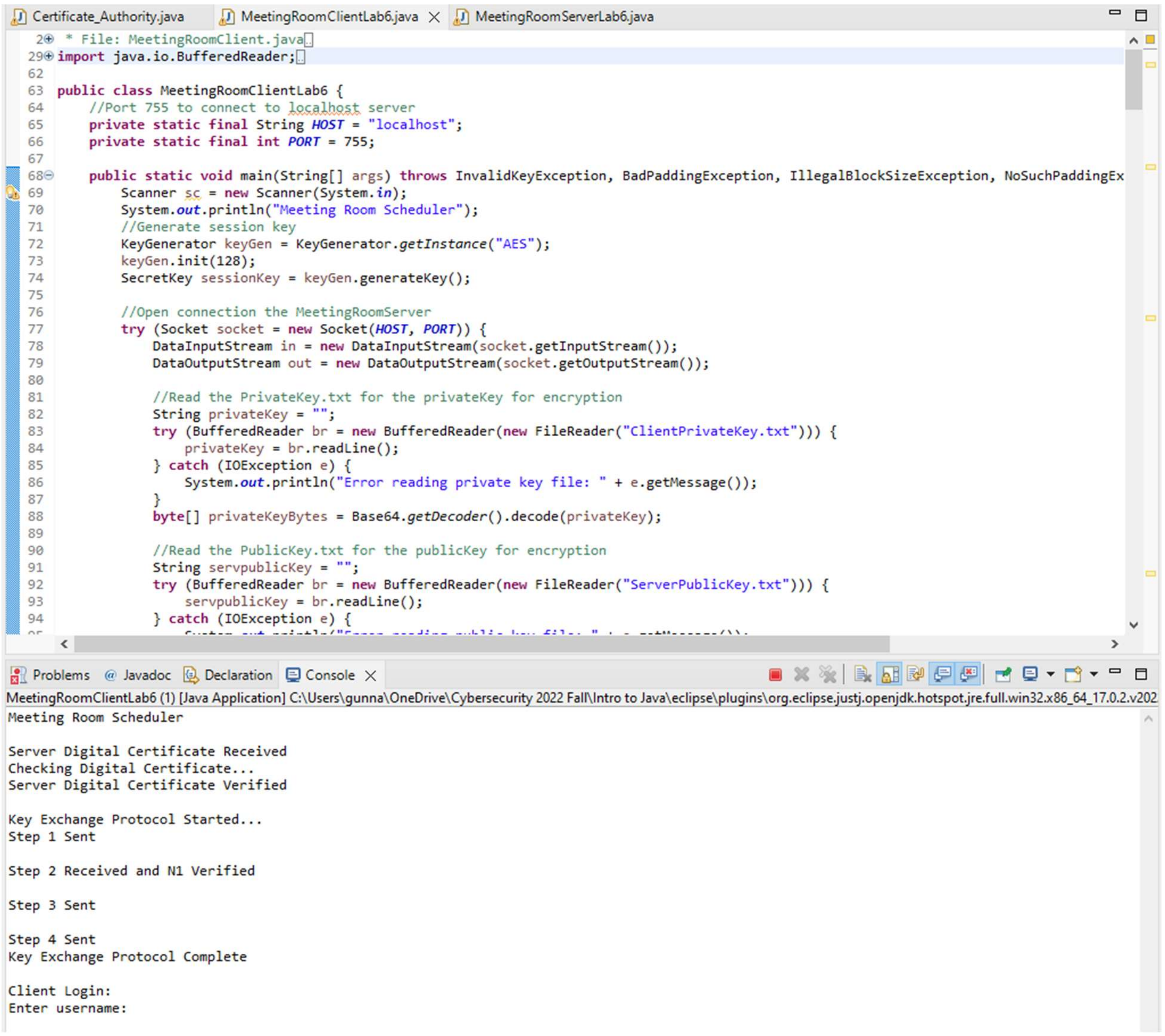

Certificate_Authority.java
MeetingRoomClientLab6.java
MeetingRoomServerLab6.java
Problems
Javadoc
Declaration
Console

```
 * File: MeetingRoomServerLab6.java
import java.io.*;

public class MeetingRoomServerLab6 {
    //Port 755 for localhost connection between client and server
    private static final int PORT = 755;
    //Text files being used that hold authorized logins and scheduled meeting times/available meeting times
    private static final String LOGIN_CREDS_FILE = "LoginCreds.txt";
    private static final String MEETING_TIMES_FILE = "MeetingTimes.txt";

    public static void main(String[] args) throws NoSuchAlgorithmException {

        while (true) {
        //Read the PublicKey.txt for the publicKey for encryption
        String publicKeyStr = "";
        try (BufferedReader br = new BufferedReader(new FileReader("ClientPublicKey.txt"))) {
            publicKeyStr = br.readLine();
        } catch (IOException e) {
            System.out.println("Error reading public key file: " + e.getMessage());
        }
        byte[] publicKeyBytes = Base64.getDecoder().decode(publicKeyStr);

        //Read the PrivateKey.txt for the privateKey for encryption
        String privateKey = "";
        try (BufferedReader br = new BufferedReader(new FileReader("ServerPrivateKey.txt"))) {
            privateKey = br.readLine();
        } catch (IOException e) {
            System.out.println("Error reading private key file: " + e.getMessage());
        }
        byte[] privateKeyBytes = Base64.getDecoder().decode(privateKey);

        //Print out that server is up and waiting for response from client
        try (ServerSocket server = new ServerSocket(PORT)) {
            System.out.println("Meeting Room Server up and running, waiting on port " + PORT + "...");
            Socket socket = server.accept();
```

```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
Meeting Room Server up and running, waiting on port 755...
```

Lab 6
Gunnar Yonker

Client view when connecting successfully:


Certificate_Authority.java
MeetingRoomClientLab6.java
MeetingRoomServerLab6.java
Problems
Javadoc
Declaration
Console

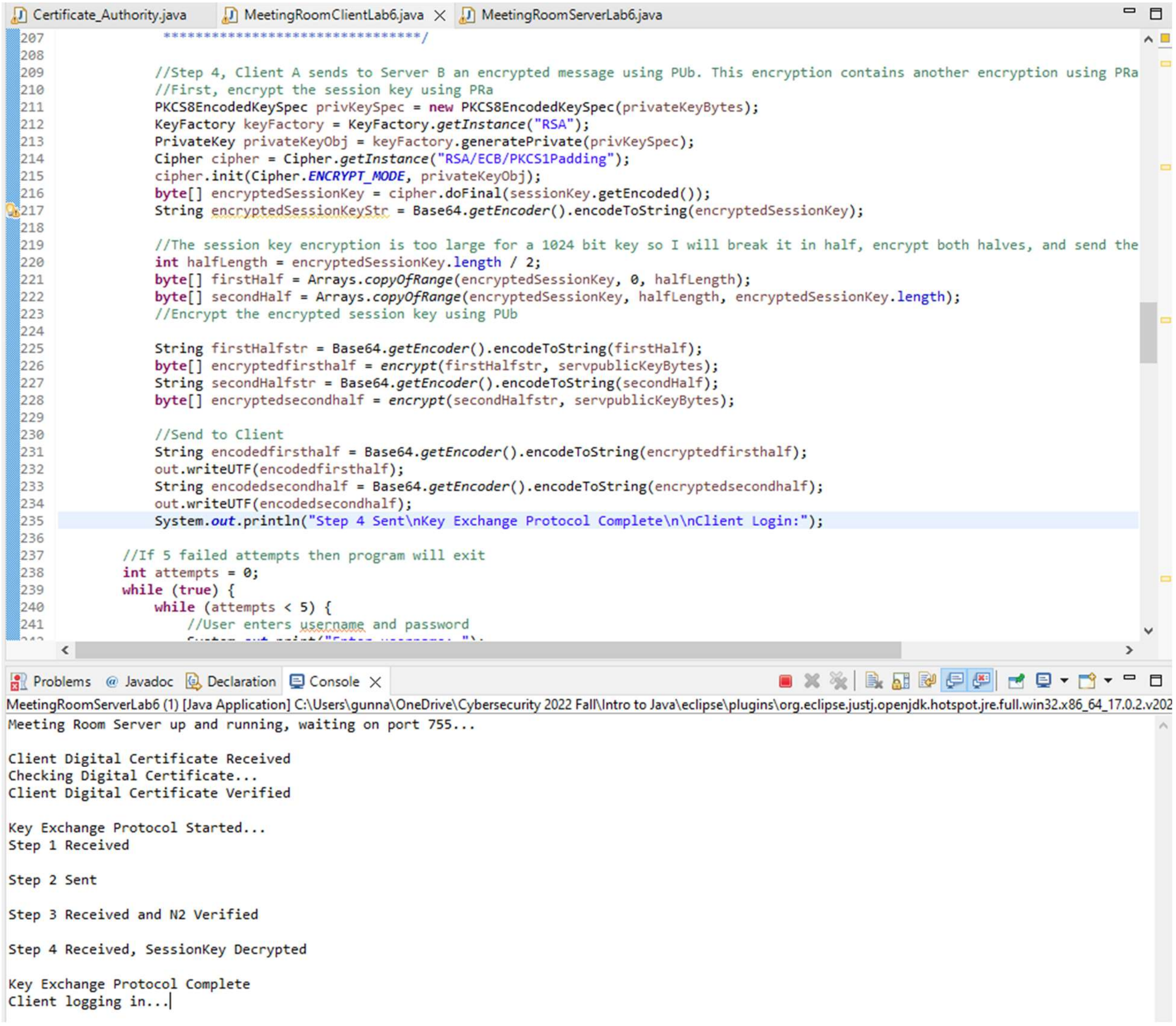
```
 * File: MeetingRoomClient.java
import java.io.BufferedReader;

public class MeetingRoomClientLab6 {
    //Port 755 to connect to localhost server
    private static final String HOST = "localhost";
    private static final int PORT = 755;

    public static void main(String[] args) throws InvalidKeyException, BadPaddingException, IllegalBlockSizeException, NoSuchPaddingEx
        Scanner sc = new Scanner(System.in);
        System.out.println("Meeting Room Scheduler");
        //Generate session key
        KeyGenerator keyGen = KeyGenerator.getInstance("AES");
        keyGen.init(128);
        SecretKey sessionKey = keyGen.generateKey();

        //Open connection the MeetingRoomServer
        try (Socket socket = new Socket(HOST, PORT)) {
            DataInputStream in = new DataInputStream(socket.getInputStream());
            DataOutputStream out = new DataOutputStream(socket.getOutputStream());

            //Read the PrivateKey.txt for the privateKey for encryption
            String privateKey = "";
            try (BufferedReader br = new BufferedReader(new FileReader("ClientPrivateKey.txt"))) {
                privateKey = br.readLine();
            } catch (IOException e) {
                System.out.println("Error reading private key file: " + e.getMessage());
            }
            byte[] privateKeyBytes = Base64.getDecoder().decode(privateKey);

            //Read the PublicKey.txt for the publicKey for encryption
            String servpublicKey = "";
            try (BufferedReader br = new BufferedReader(new FileReader("ServerPublicKey.txt"))) {
                servpublicKey = br.readLine();
            } catch (IOException e) {
```

```
MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
Meeting Room Scheduler

Server Digital Certificate Received
Checking Digital Certificate...
Server Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Sent

Step 2 Received and N1 Verified

Step 3 Sent

Step 4 Sent
Key Exchange Protocol Complete

Client Login:
Enter username:
```

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.

Lab 6
Gunnar Yonker

Server view:



Certificate_Authority.java
MeetingRoomClientLab6.java
MeetingRoomServerLab6.java
Problems
Javadoc
Declaration
Console

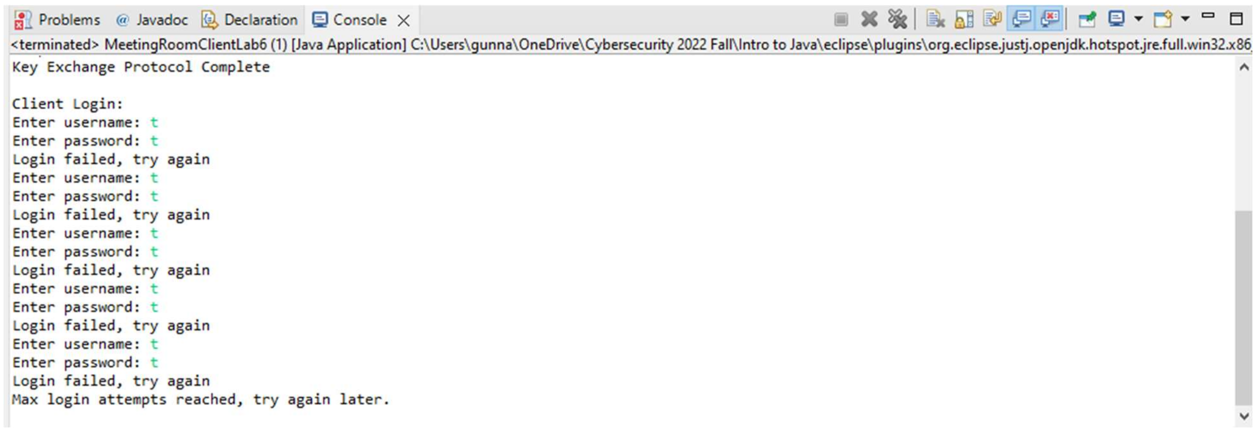
```java
            *******************************/

            //Step 4, Client A sends to Server B an encrypted message using PUb. This encryption contains another encryption using PRa
            //First, encrypt the session key using PRa
            PKCS8EncodedKeySpec privKeySpec = new PKCS8EncodedKeySpec(privateKeyBytes);
            KeyFactory keyFactory = KeyFactory.getInstance("RSA");
            PrivateKey privateKeyObj = keyFactory.generatePrivate(privKeySpec);
            Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
            cipher.init(Cipher.ENCRYPT_MODE, privateKeyObj);
            byte[] encryptedSessionKey = cipher.doFinal(sessionKey.getEncoded());
            String encryptedSessionKeyStr = Base64.getEncoder().encodeToString(encryptedSessionKey);

            //The session key encryption is too large for a 1024 bit key so I will break it in half, encrypt both halves, and send the
            int halfLength = encryptedSessionKey.length / 2;
            byte[] firstHalf = Arrays.copyOfRange(encryptedSessionKey, 0, halfLength);
            byte[] secondHalf = Arrays.copyOfRange(encryptedSessionKey, halfLength, encryptedSessionKey.length);
            //Encrypt the encrypted session key using PUb

            String firstHalfstr = Base64.getEncoder().encodeToString(firstHalf);
            byte[] encryptedfirsthalf = encrypt(firstHalfstr, servpublicKeyBytes);
            String secondHalfstr = Base64.getEncoder().encodeToString(secondHalf);
            byte[] encryptedsecondhalf = encrypt(secondHalfstr, servpublicKeyBytes);

            //Send to Client
            String encodedfirsthalf = Base64.getEncoder().encodeToString(encryptedfirsthalf);
            out.writeUTF(encodedfirsthalf);
            String encodedsecondhalf = Base64.getEncoder().encodeToString(encryptedsecondhalf);
            out.writeUTF(encodedsecondhalf);
            System.out.println("Step 4 Sent\nKey Exchange Protocol Complete\n\nClient Login:");

        //If 5 failed attempts then program will exit
        int attempts = 0;
        while (true) {
            while (attempts < 5) {
                //User enters username and password
```

```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
Meeting Room Server up and running, waiting on port 755...

Client Digital Certificate Received
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...
```

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.

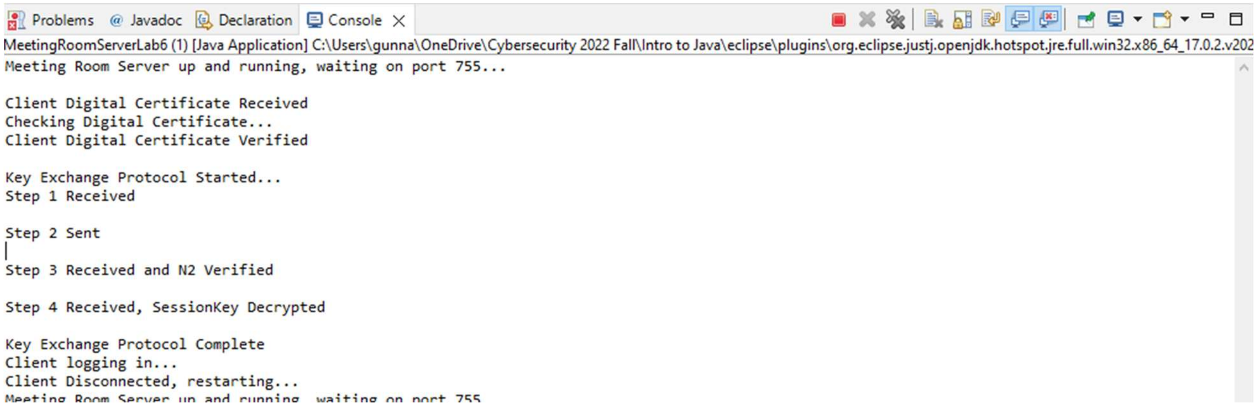
Lab 6
Gunnar Yonker

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



```
<terminated> MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86
Key Exchange Protocol Complete

Client Login:
Enter username: t
Enter password: t
Login failed, try again
Enter username: t
Enter password: t
Login failed, try again
Enter username: t
Enter password: t
Login failed, try again
Enter username: t
Enter password: t
Login failed, try again
Enter username: t
Enter password: t
Login failed, try again
Max login attempts reached, try again later.
```

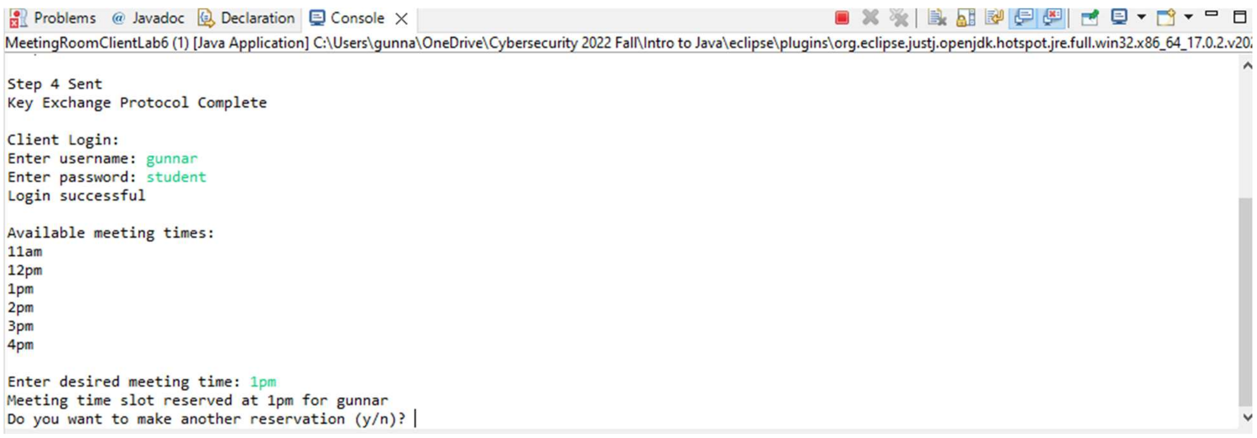
Server view:



```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
Meeting Room Server up and running, waiting on port 755...

Client Digital Certificate Received
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...
Client Disconnected, restarting...
Meeting Room Server up and running, waiting on port 755
```

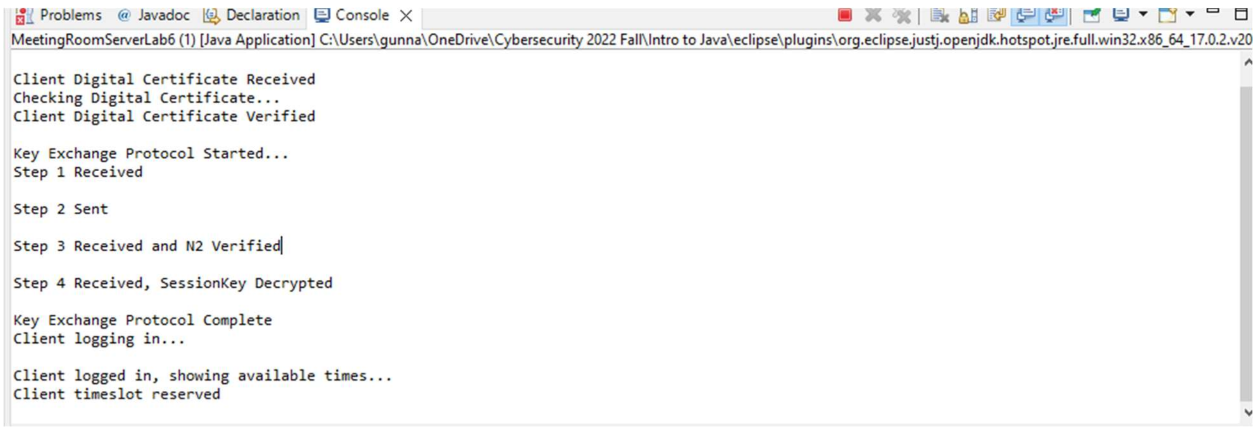
Lab 6
Gunnar Yonker

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



```
MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20

Step 4 Sent
Key Exchange Protocol Complete

Client Login:
Enter username: gunnar
Enter password: student
Login successful

Available meeting times:
11am
12pm
1pm
2pm
3pm
4pm

Enter desired meeting time: 1pm
Meeting time slot reserved at 1pm for gunnar
Do you want to make another reservation (y/n)?
```

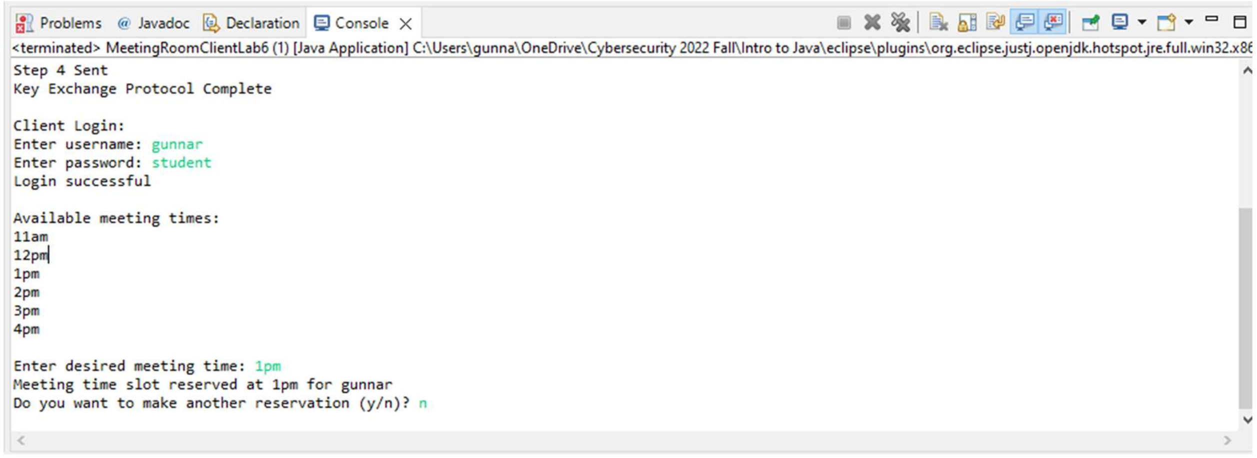
Server view:



```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20

Client Digital Certificate Received
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...

Client logged in, showing available times...
Client timeslot reserved
```

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

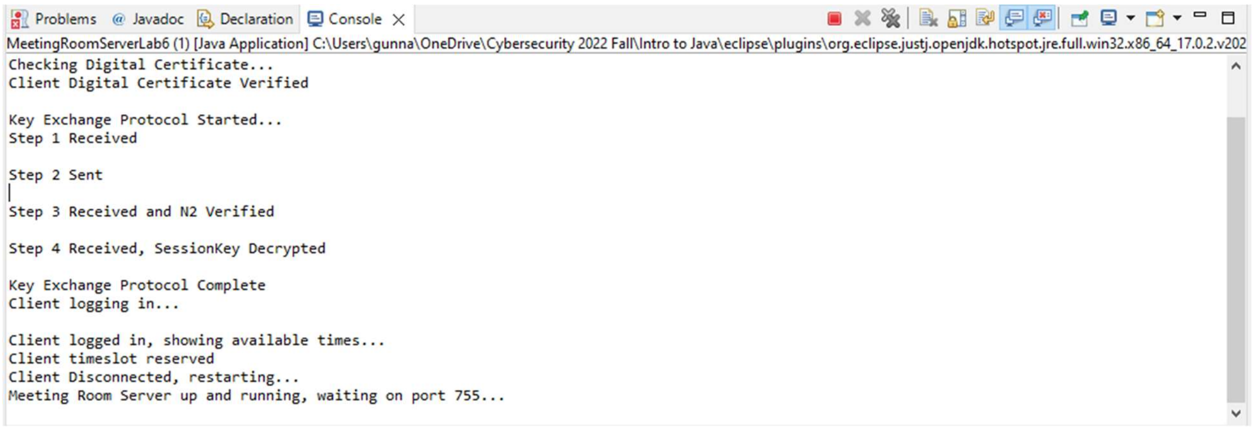
Client view:



```
<terminated> MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86
Step 4 Sent
Key Exchange Protocol Complete

Client Login:
Enter username: gunnar
Enter password: student
Login successful

Available meeting times:
11am
12pm
1pm
2pm
3pm
4pm

Enter desired meeting time: 1pm
Meeting time slot reserved at 1pm for gunnar
Do you want to make another reservation (y/n)? n
```

Lab 6
Gunnar Yonker

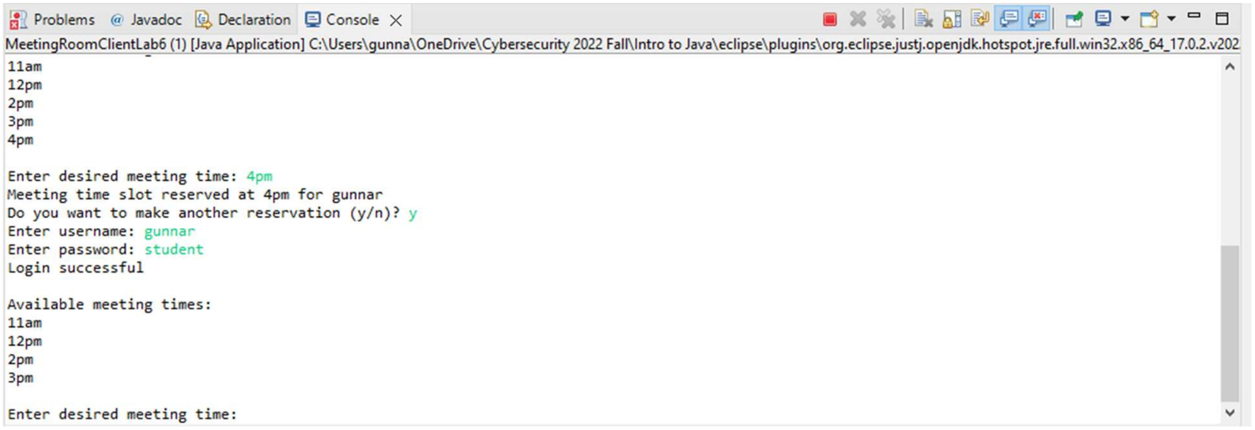
Server view:



```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...

Client logged in, showing available times...
Client timeslot reserved
Client Disconnected, restarting...
Meeting Room Server up and running, waiting on port 755...
```

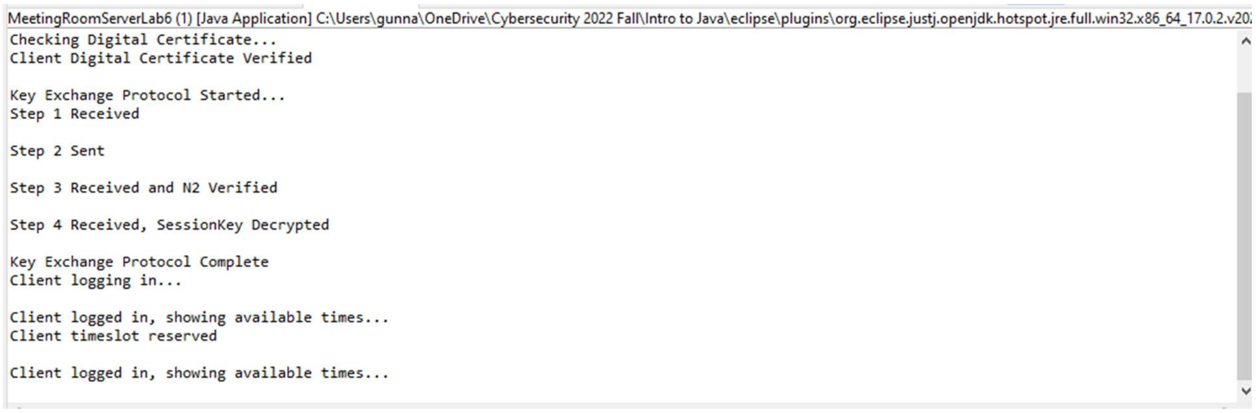
If client says y, then they are prompted to login and make another reservation:

Client view:



```
MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v202
11am
12pm
2pm
3pm
4pm

Enter desired meeting time: 4pm
Meeting time slot reserved at 4pm for gunnar
Do you want to make another reservation (y/n)? y
Enter username: gunnar
Enter password: student
Login successful

Available meeting times:
11am
12pm
2pm
3pm

Enter desired meeting time:
```

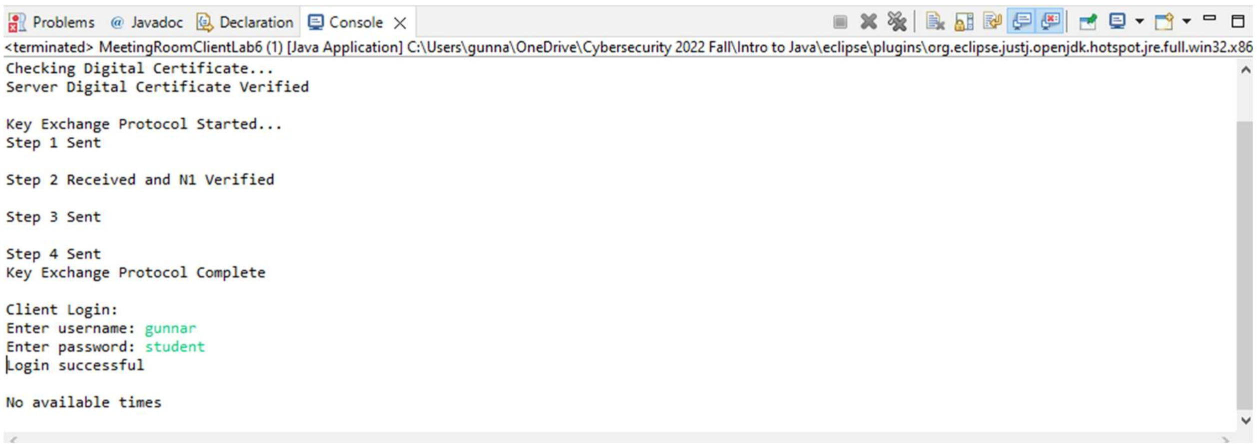
Server view:



```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...

Client logged in, showing available times...
Client timeslot reserved

Client logged in, showing available times...
```

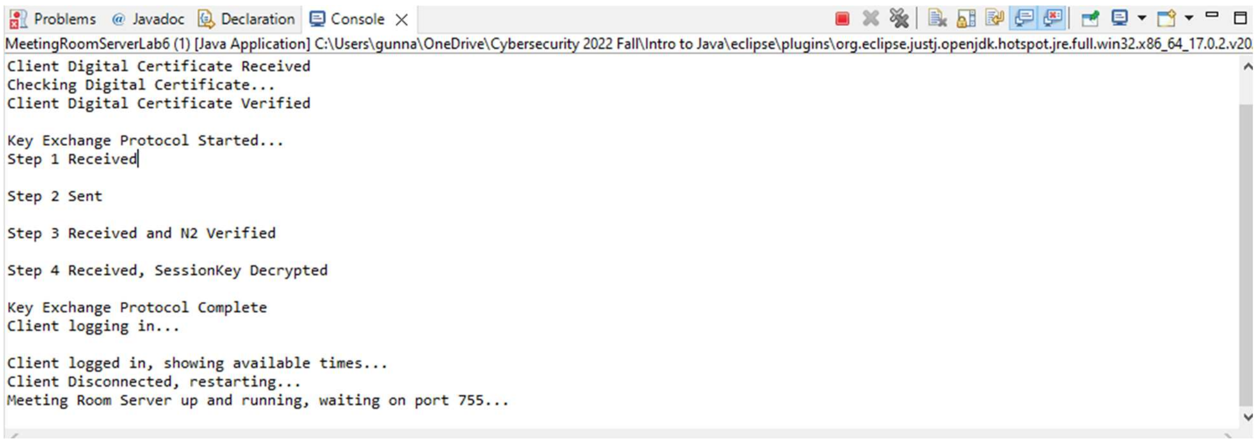
Lab 6
Gunnar Yonker

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



```
<terminated> MeetingRoomClientLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86
Checking Digital Certificate...
Server Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Sent

Step 2 Received and N1 Verified

Step 3 Sent

Step 4 Sent
Key Exchange Protocol Complete

Client Login:
Enter username: gunnar
Enter password: student
Login successful

No available times
```

Server view:



```
MeetingRoomServerLab6 (1) [Java Application] C:\Users\gunna\OneDrive\Cybersecurity 2022 Fall\Intro to Java\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20
Client Digital Certificate Received
Checking Digital Certificate...
Client Digital Certificate Verified

Key Exchange Protocol Started...
Step 1 Received

Step 2 Sent

Step 3 Received and N2 Verified

Step 4 Received, SessionKey Decrypted

Key Exchange Protocol Complete
Client logging in...

Client logged in, showing available times...
Client Disconnected, restarting...
Meeting Room Server up and running, waiting on port 755...
```