COSC 450 Assignment 1 Part 2

Group: Mike Santoro, Ryan Fongheiser, Steve Smith, Cesar Suchite, Eric Louhi

README

Our TCP socket program to perform Diffie-Hellman key exchange consists of 2 .java files, one for the server and one for the client (Server.java and client\_display.java respectively). First we will discuss the general logic of the program, and following that we will provide instructions for testing.

To briefly touch on the server and socket part, which is a modification from a server socket program used for COSC 350. First we setup the server socket on port 444 and have it listening for clients. The server program prints the port it is running on and the ip address as well, so that it is easy for the client to see and connect. Once the client program is run, you will use this ip and port number, type them into the appropriate input boxes and press connect. The server will accept, bind, and now the client will be connected to the server. Our program sends hello messages back and forth to confirm connection. So once connected, the client should type into the message box, three integers separated by spaces, The prime number (p), the primitive root g modulo p, and the secret key a. After selecting send message, the client program will take the message entered in the dialogue box , split it up into an array of strings, convert those to the appropriate integer variables, and then again convert them to BigInteger objects. We use BigInteger so that we can use the modPow() function for easy arithmetic. So the client takes p, g, and a...calculates g^a mod p to get public key A, then convert back to int then String, and combines p, g, and A to send as one string message to the server. The client is then left listening for a message response from the server.

Once the server receives this string, it goes through the same parsing process and converts to int then BigInteger. It then prompts the user to choose and enter a secret or private key b. This is converted to BigInteger and then used to calculate the public key B. Once we have the public key B, the server sends this value back to the client so it can be used to compute the shared DH secret key. The server then computes this key using the public key A sent from the client. Using s = A^b mod p the server calculates and prints the shared secret key.

The client receives the public key B, and calculates the shared private key using equation s = B^a mod p, and also prints this to the screen. This completes the Diffie-Hellman key exchange.

**Testing the Program**

To test the program perform the following steps

* Download the Server.java and client\_display.java files. Open them in a java IDE (we used netbeans to develop, however dr. java, eclipse, etc. will also work)
* Save and compile both of the files.
* First, run the Server.java file, this should open the server and socket while also printing the ip address and port number that will be needed for the client.
* Next run the client\_display.java file. A GUI should appear.
* Enter the given IP address and port number of the client in the appropriate boxes and press connect. You should receive confirmation of a successful connection.
* Enter the prime number, primitive root, and secret key a in the message box, and select “send\_message” button.
* Next the server console should prompt for a secret key b. Enter this secret key and hit enter.
* Finally both the server and the client should print the same shared secret key. If the same the program has worked correctly.

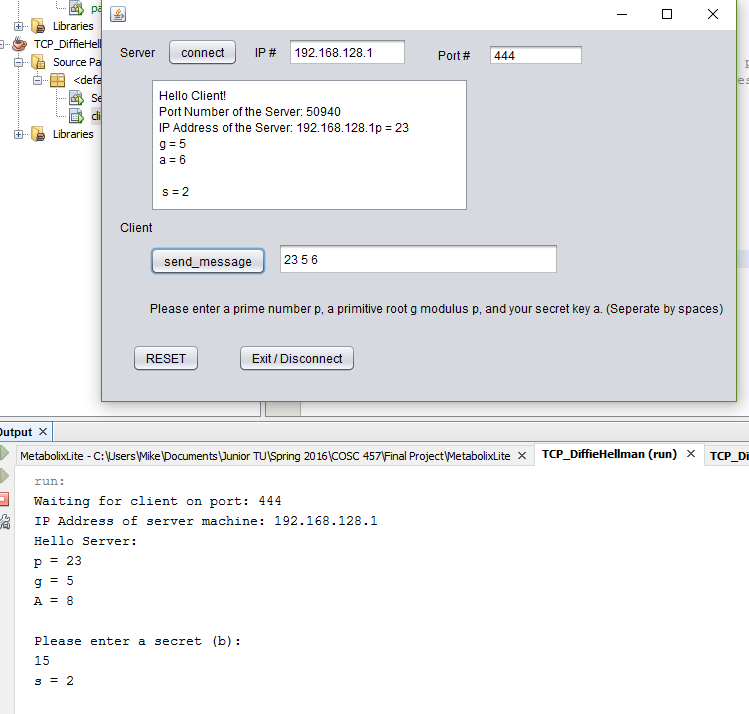
\*These instructions allow testing of both server and client being run on the local machine. It can also be tested between two different boxes, just make sure you are on the same open network.

**Resources Used**

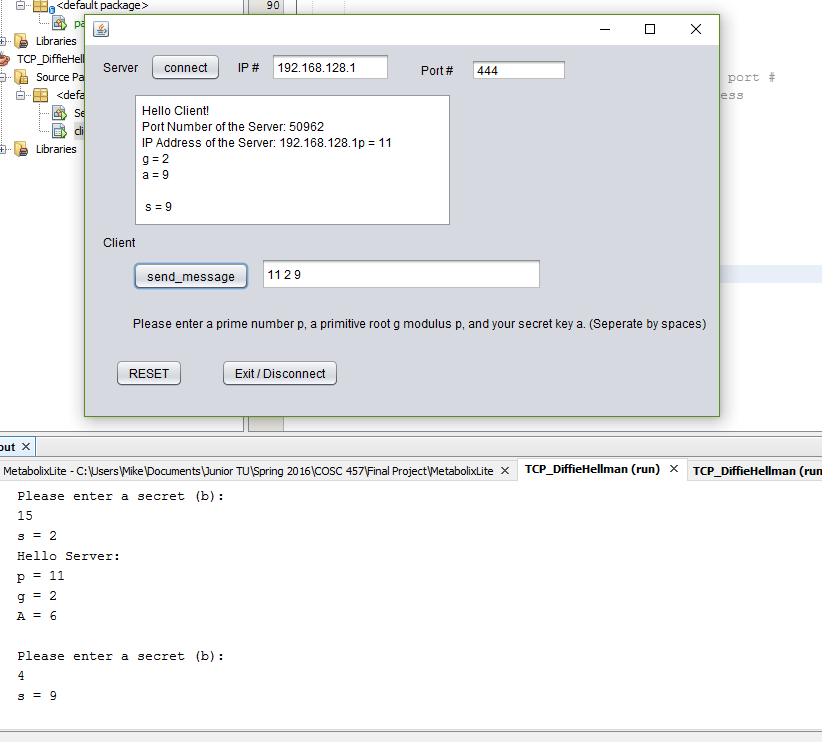
* [**http://mathworld.wolfram.com/Diffie-HellmanProtocol.html**](http://mathworld.wolfram.com/Diffie-HellmanProtocol.html)
* [**http://www.math.ucla.edu/~baker/40/handouts/rev\_DH/node1.html**](http://www.math.ucla.edu/~baker/40/handouts/rev_DH/node1.html)

**Screen Shots**

* Example input (p = 23, g = 5, a = 6, b = 15)



* Example input (p = 11, g = 2, a = 9, b = 4)



**Limitations**

* A few additions we could have made include the programs being able to generate the prime and primitive root on their own, as of now the program relies on the user to understand diffie-hellman to enter the correct numbers.