ISA 656 - Program Manual

Krishnan Subramanian
G00689208

Instructions to run the program

- The following files should be found:
  - Client.java
  - Server.Java
  - ClientAuth.java
  - ServerAuth.java
  - server.key
  - krish.key
- Please compile all the programs first before execution. Java compiler jre version 1.6+ preferred
- After compilation, to run the server program:

  ```
  java Server <port> <serverkeyfile path>
  ```

  Note: If the Server's .key file resides in the same directory as the class files, just specify only the .key file name, otherwise include the whole path of the file
- to run the client program:

  ```
  java Client <server hostname/IP> <port> <username> <clientkeyfile path>
  ```

  Note: If the Client's .key file resides in the same directory as the class files, just specify only the .key file name, otherwise include the whole path of the file
- If the client is running on the a different machine across the network (i.e. not on the localhost), you will have to copy the krish.key file and the two java files (Client.java & ClientAuth.java) for the program to run properly.
- The program also supports multiple clients and multithreaded environment

How the program works

- First, the client looks for the server's public key entry in it's .key keystore file (`krish.key`).
- It then encrypts the username using the server's public key and sends it over to the server.

- The server looks for its .key keystore file i.e.(`server.key`)
- It decrypts the username sent by the client using its private key stored in the keystore file
- If it finds a match, then it sends a status message `200` to the client, otherwise a status message of `400` is sent
- The client receives the response message from the server and notifies the user if the authentication was successful/failed.
- If the authentication was successful with the server, it generates a new DES key for the session
- The generated DES key is then sent over to the server using the server's public key
- The server decrypts the DES key and notifies the client with a status message `200` to notify that the Encrypted Key Exchange (EKE) was successful
- The server uses this DES key for encrypting all future messages till the session is closed

Creating your own keystore

Please follow these steps if would would like to create your own keystores for the server and client

- To create a keystore use the following command:

  ```
  keytool -genkey -alias server -keyalg RSA -keystore .serverstore
  ```

  This would by default create & add an a RSA key pair with the alias "server"

- To list the keys bound in the keystore:

  ```
  keytool -list -keystore .serverstore
  ```

- To export the key as a public certificate:

  ```
  keytool -export -alias krish -file pubkrish.cer -keystore krish.key
  ```

  This would create a file called "pubkrish.cer" on the current directory

- To import the key as a public certificate:

  ```
  keytool -import -alias pub_server -file pubserver.cer -keystore krish.key
  ```

Please see the file keycommands.txt if you would like to directly copy these commands

Class & File Descriptions

- **Server.java**: The main multithreaded chat server program
- **Client.java**: The client program
- **ServerAuth.java**: Contains all authentication functions used by the server
- **ClientAuth.java**: Contains all authentication functions used by the client
- **server.key**: The keystore file used by the server program
- **krish.key**: The keystore used by the client program

Please see the generated javadoc folder for more information - doc