Secure Messenger Documentation

A secure instant messenger between client and server, written in Java

This project was built for SENG 360 (Security Engineering) at the University of Victoria.
The full assignment requirements can be found in `requirements.pdf`.
For extensive method documentation, see the `/docs/` folder.
The code was written by Kyle Thorpe and Daniel Frankcom.

Instructions: Please note, picture instructions are at the end of this document.

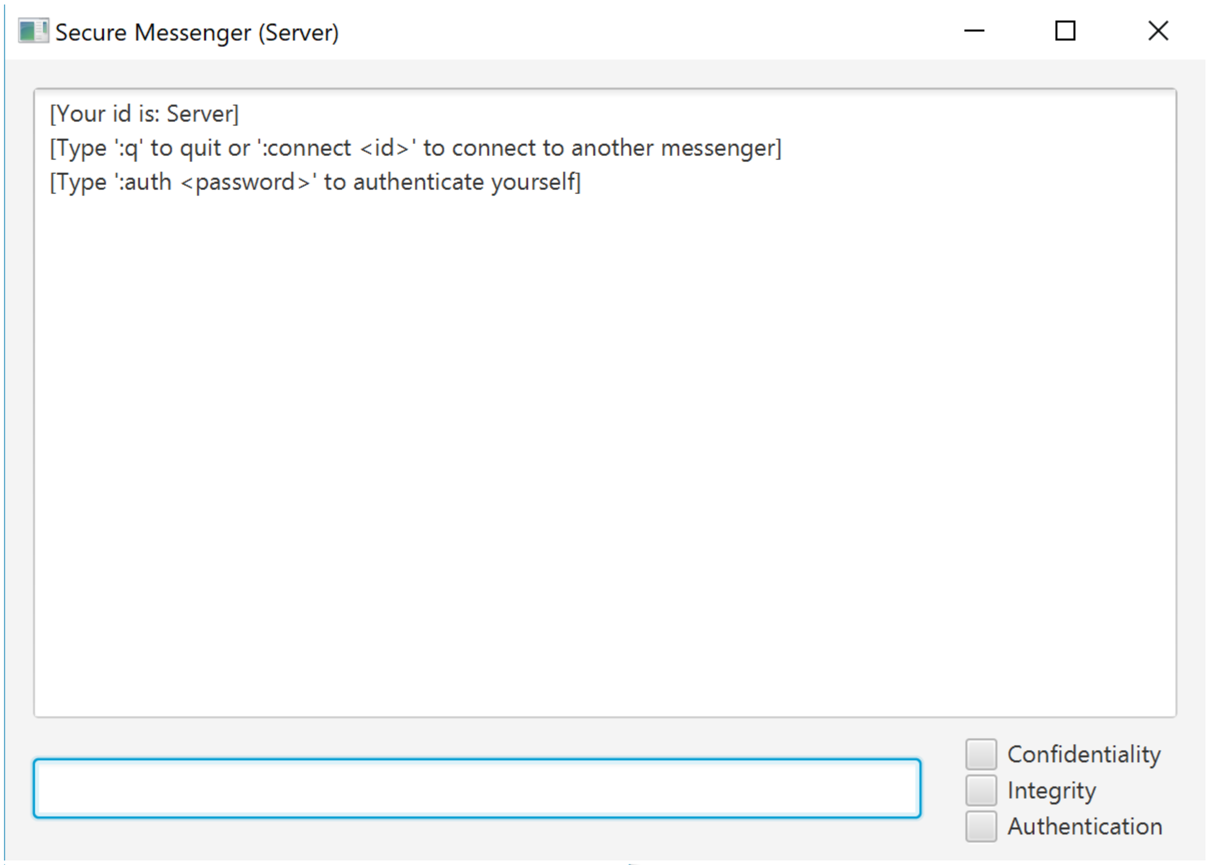
1. To run the code, first compile with `javac *.java`
2. Next, run a server with `java Messenger server`
3. Select the required security parameters on the right of the textbox. The available options are Confidentiality, Integrity, and Authentication.
4. Type :connect client in the GUI
   If authentication is required, the password is `hunter2`
   Type in the gui `:auth hunter2`
5. Finally, run a client with `java Messenger client`
6. The security parameters must be the same for both client and server, or else a connection cannot be made.
7. Type :connect server
8. If authentication is required, the password is `hunter3`
   Type in the gui `:auth hunter3`
9. To disconnect from the current session, type :disconnect

Follow the prompts in the GUI for connection and communication

Many design decisions were made while working on this project, here are some of them:

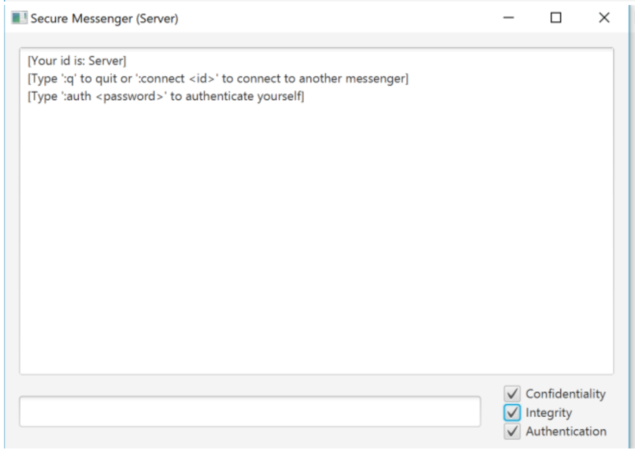
- The security factor of Integrity is implemented using a hash which is encrypted under the receiver's public key (using RSA). The receiver will decrypt this using their private key, and check to make sure the message hashes to the same bytes.
- Authentication is done by accepting a password from the command :auth. This plaintext password is then hashed using MD5, and is compared to the password hash stored in the file "users".
- Communication between the client and server is done using RMI.
- We chose to not allow a user to receive a message unless they have authenticated (If Authentication is a required security parameter). This will prevent an attacker from impersonating someone and gaining confidential information.

- We chose to force a user to reauthenticate if they leave a connection and then reconnect. This will prevent an attacker from accessing confidential information when Secure Messenger is left open on a victim's computer.


Secure Messenger (Server)

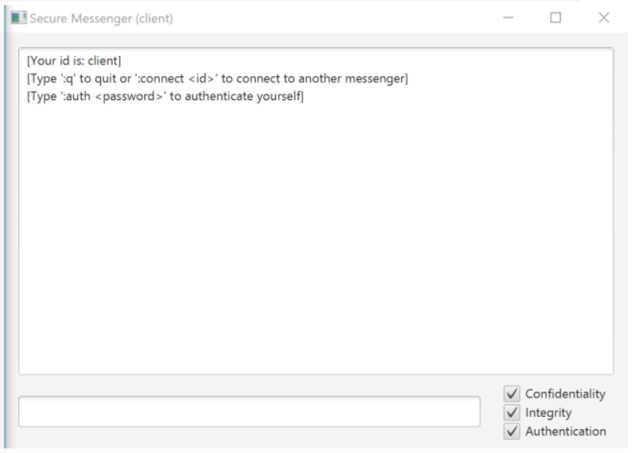
```
[Your id is: Server]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
```

☐ Confidentiality
☐ Integrity
☐ Authentication


Secure Messenger (Server)

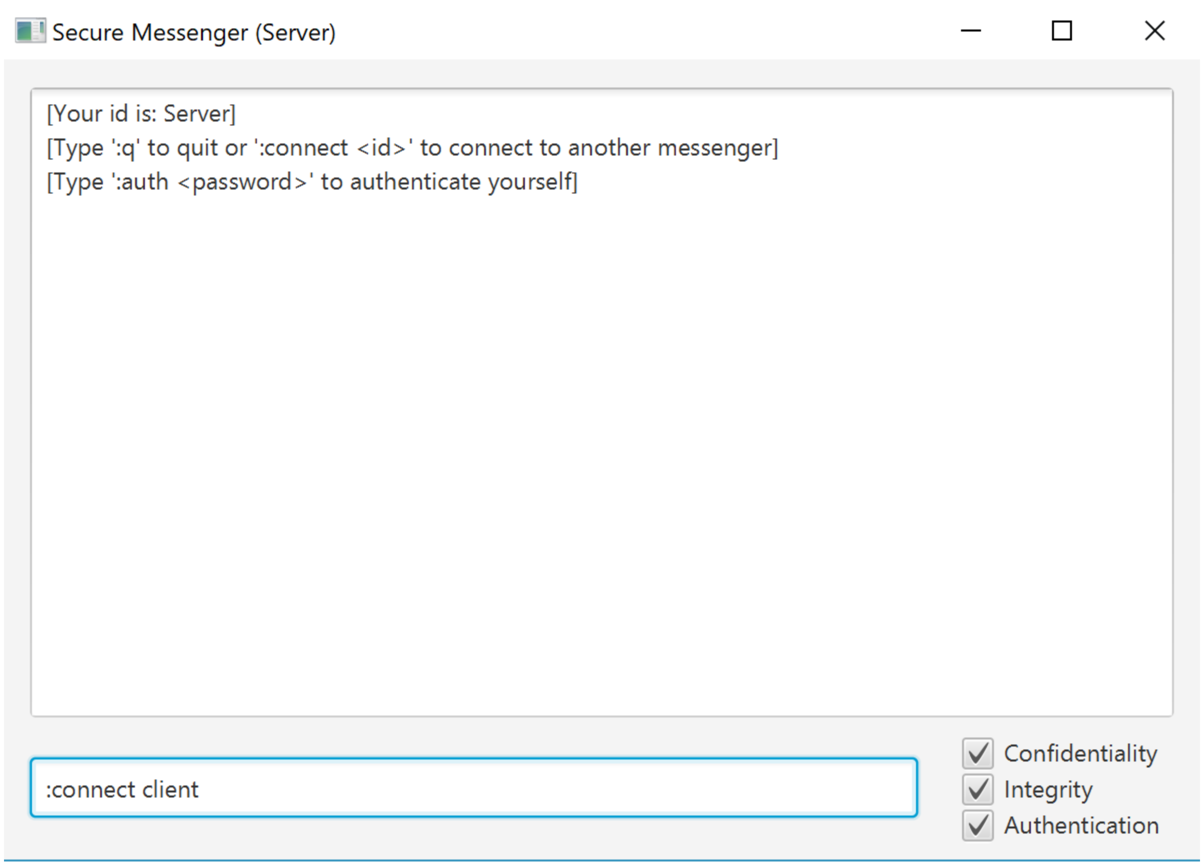
```
[Your id is: Server]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
```

✓ Confidentiality
✓ Integrity
✓ Authentication


Secure Messenger (client)

```
[Your id is: client]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
```

✓ Confidentiality
✓ Integrity
✓ Authentication

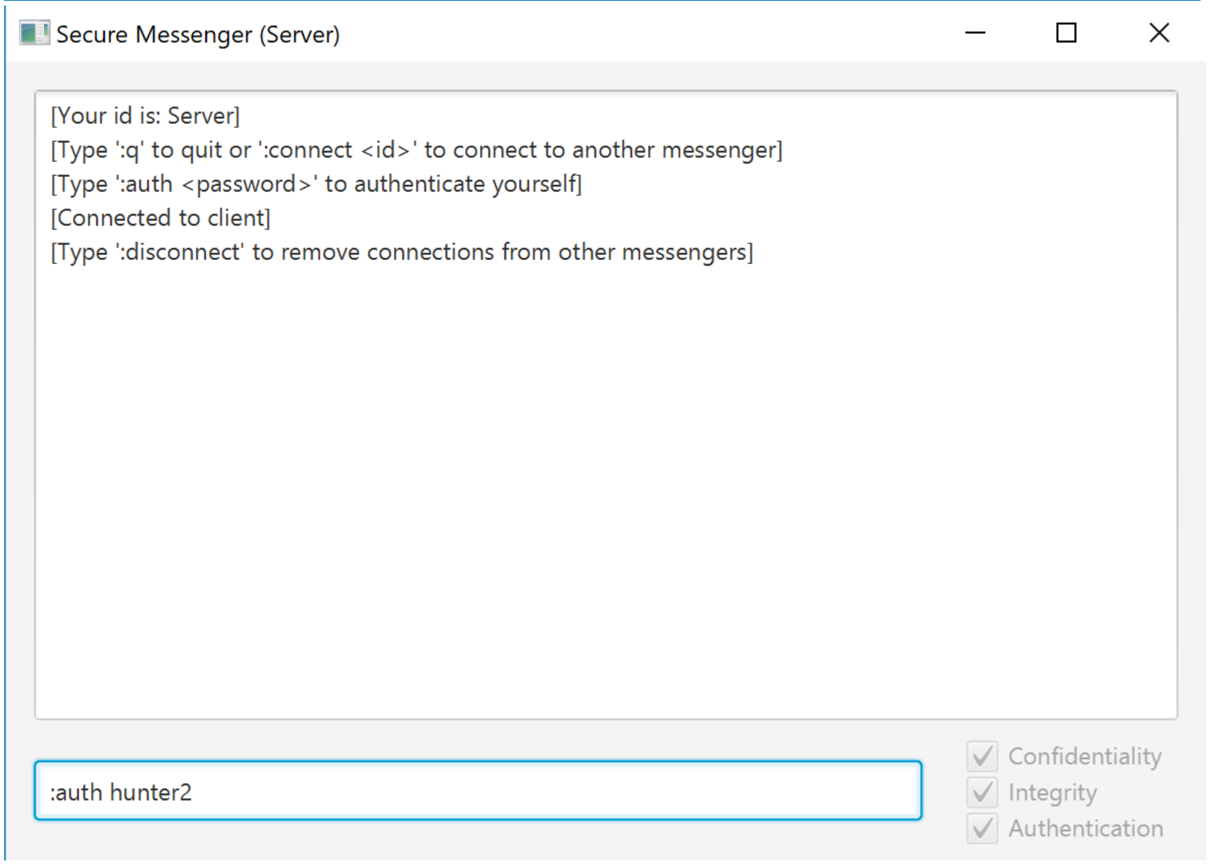


Secure Messenger (Server)

```
[Your id is: Server]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
```

```
:connect client
```

- ✓ Confidentiality
- ✓ Integrity
- ✓ Authentication

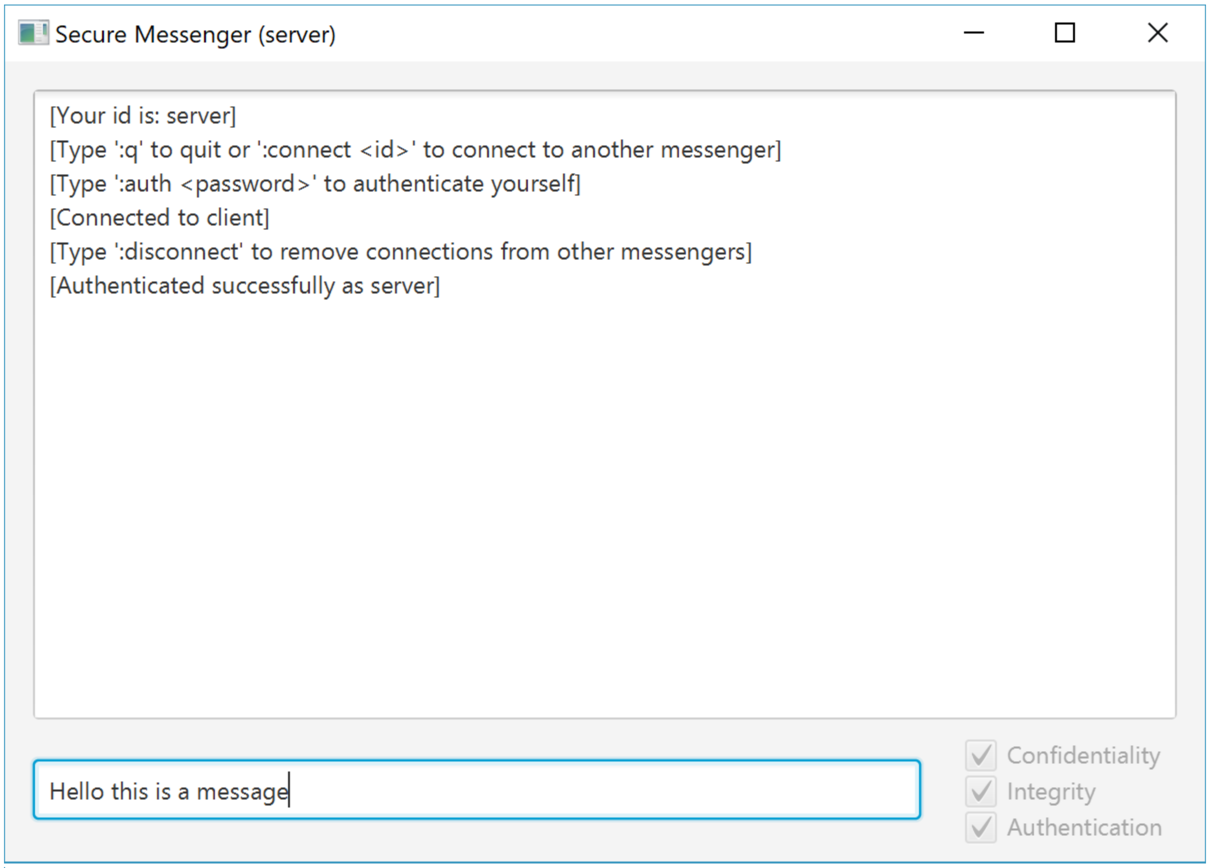


Secure Messenger (Server)

```
[Your id is: Server]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
[Connected to client]
[Type ':disconnect' to remove connections from other messengers]
```

```
:auth hunter2
```

- ✓ Confidentiality
- ✓ Integrity
- ✓ Authentication

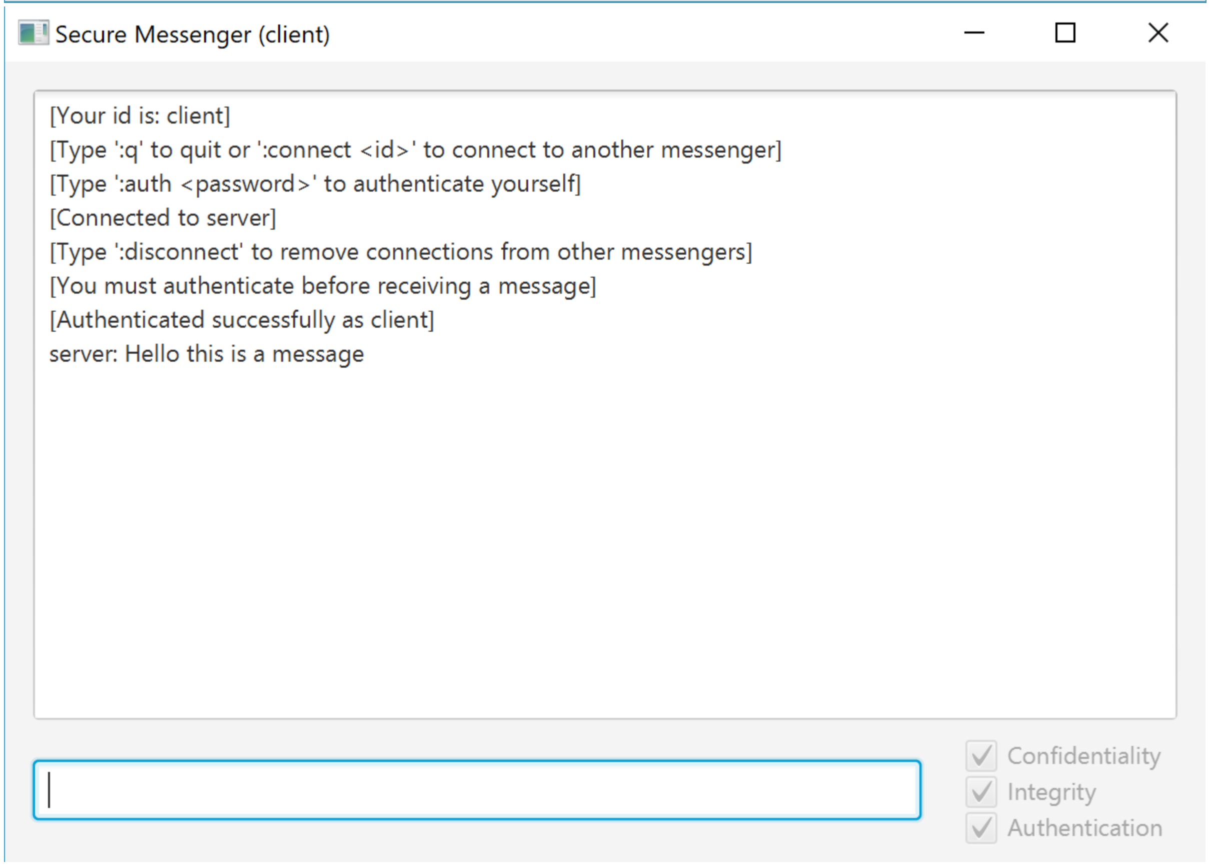


Secure Messenger (server)

```
[Your id is: server]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
[Connected to client]
[Type ':disconnect' to remove connections from other messengers]
[Authenticated successfully as server]
```

Hello this is a message

✓ Confidentiality
✓ Integrity
✓ Authentication



Secure Messenger (client)

```
[Your id is: client]
[Type ':q' to quit or ':connect <id>' to connect to another messenger]
[Type ':auth <password>' to authenticate yourself]
[Connected to server]
[Type ':disconnect' to remove connections from other messengers]
[You must authenticate before receiving a message]
[Authenticated successfully as client]
server: Hello this is a message
```

✓ Confidentiality
✓ Integrity
✓ Authentication