# SENG 360 Assignment 3 - Cryptography

James Barlow
Douglas Hon
Bryce McEwan

## Technical Details

### General

Our project utilizes the Java Cryptographic Architecture. Handshakes are used on initial startup to confirm that the mode the server is currently running in is the same as the client.

### Confidentiality

Confidentiality is implemented using the AES symmetric encryption standard with a PKCS5 padding scheme. Our program uses a Diffie-Hellman key exchange to generate a 2048 bit key for our encryption. On start up of the Server or Client, the program generates its own public key. This is shared when a connection between the Server and Client is established. The program generates all of the needed parts for the creation of the key. Diffie-Hellman allows the Server and Client to generate the same key using their own private key, and a shared secret that was created by the other party.

### Integrity

To ensure integrity, our program uses digital signatures. We use the Java Cryptographic Library to create a public and private key. If integrity is enabled, these keys will be generated automatically and will be used in the creation of the digital signatures and the checking of the integrity of the message.

### Authentication

Authentication is accomplished through handshakes between the client and the server. We will cover here what is sent in which direction and the steps taken by each party outside of just sending the message itself.

When authentication is selected client-side the user is to enter 3 pieces of information:

- Username
- Password
- Secret

The username and password are then concatenated space separated and sent off to the server which then separates them, hashes them, encodes them with Base64 encoding, and then checks in the list of authenticated users for a matching hashed username and password. If it finds a match the server then sends back the secret associated with the username and password to the client-side application. Note that this secret is ciphered with a key only known by the client-side application and then encoded in Base64, and was passed out of band to the server's authenticated users list. The client then decodes and then decrypts the secret sent to it by the server and check for a match with what was entered by the user earlier. If it matches the client send a message back to the server saying that it accepted the authentication and then goes into normal chat mode. Upon reception of the message from the client the server does the same.
Some more technical detail with regards to hashing, and ciphers used is as follows:

- Hashing Algorithm is SHA-256
- Cipher is AES/ECB/PKCS5Padding
- Encoding is all Base64

# Compiling the Program

Our program uses Java files, and can be compiled using the following commands:

```
javac Client.java
javac Common.java
javac MessageListener.java
javac Server.java
```

# Using the Program

`Server.java` must be running before you run `Client.java`.

`Server.java` can be started using the following commands:

```
java Server <host name> <port number> <security option>
```

The possible security options are:

| | |
|---|---|
| C | Enables Confidentiality |
| I | Enables Integrity |
| A | Enables Authentication |

These flags can be used in any combination to enable different varieties of security options. For example, to run a server with the host name ‘localhost’, the port number 18080, with the security options of Confidentiality and Integrity, you would use the command:

```
java Server localhost 18080 CI
```

`Client.java` can be started using the following commands:

```
java Client <port number> <security option>
```

The possible security options are:

| | |
|---|---|
| C | Enables Confidentiality |
| I | Enables Integrity |
| A | Enables Authentication |

These flags can be used in any combination to enable different varieties of security options. For example, to run a client on the port number 18080, with the security options of Integrity and Authentication, you would use the command:

```
java Client 18080 IA
```

The server currently has three authenticated users: bmcewan, jbarlow, and dhon. All with passwords and secrets of password and secret respectively. These accounts can be used in testing out the authentication capabilities for marking.