**Encryption**

The external library *CryptoSwift* is used for AES encryption, decryption, padding, block chaining as well as HMAC generation. In regards to padding; from the CryptoSwift GitHub page:

*“Manual padding of data is optional,*

*and CryptoSwift is using PKCS7 padding by default.“*

Plaintext is encrypted using:

AES-256

CBC (16-byte IV) Block-Mode

PKCS7 Padding

RSA encryption and decryption of the concatenated AES and HMAC key is done through *SwCrypt* external library. The application bundle will generate a 2048-bit DER formatted key pair when executing its first encryption. This happens through *SwCrypt* as well:

CC.RSA.generateKeyPair(2048)

To-Do: This pair are stored in keychain.

Cryptographically secure numbers are generated using the default Apple framework *Security* under *Randomization Services.*

Output from the encryption module is in JSON format. It follows the following structure:

{

"keys-cipherText": “value”,

"content-cipherText": “value”,

"hmac": “value”,

"iv": “value”,

}

*“value”s* are strings encoded in Base64.

HMAC is generated on the ciphertext – not plaintext.

**Repository Management**

The Node server responsible for responding to API calls is located at:

~/messengerAPI/

This directory is a bare git repository (git init --bare). Pushing to GitHub remote ‘origin’ master branch will also initiate a push to the AWS remote production branch by a git hook:

git push production master

Upon receiving the update, the server will restart the node server.js file responsible for responding to API calls:

sudo pm2 restart server

If the API returns a 502 Bad Gateway after a commit, there was probably an error was probably encountered server-side. In this case, log into the the server using SSH described in **Logging into AWS Server**. Once logged in, run the following command:

sudo pm2 list

The process named *server* would show “errored” in the status column to indicate there was something wrong in the codebase. To view the error, use the following command:

sudo pm2 logs server

**Logging into AWS Server**

Login happens by SSH under the default user ‘ubuntu’. The connection IP is shown below. Development client computers have SSH keys matched with the server.

ssh ubuntu@ec2-13-58-143-176.us-east-2.compute.amazonaws.com

**External Libraries**

Most client-side external libraries have been imported using the *CocoaPods* system. Updating the file *Podfile* with a new pod located in the client project folder, requires calling the following command on the project folder directory:

pod update

TODO:

Update server.js with JWT functionality

**Message API**

Route: ‘/messages’

GET

Returns all messages.

POST

Upload a new message.

content: String

upload\_date: Date

NYI Route: ‘/login’

NYI Route: ‘/signup’

**Server Setup**

Ubuntu 14

Nginx

MongoDB & Mongoose

PM2

NPM

Node.js

Express

AngularJS