Ariele Silva

MSCS630

Prof. Rivas

10 May 2020

Writeup

This semester my project was to create a phone app that will allow the user to encrypt and decrypt plaintext in RSA. I had intended for this app to be created for the iphone, but due to Covid-19, and Marist evacuating the students off campus, I no longer had access to the Mac Labs, to create the iOS based application. As an alternative, I had to switch my OS and create the app in Android Studio rather than Xcode.

During this process of switching over, I had to recreate what I had already started for my application. I was in the starting stages for my app, so at the time, all I had created in Xcode was the structure of how my app would look. The wireframe of the iOS app was created using Swift. I had begun to create the button functions and view sets for the text inputs when we had to leave marist, which caused me to then have to recreate all this again in Java for Android Studio. Unfortunately, losing access to the Mac Lab is what caused me to have to switch my OS since I don't own a Mac computer / laptop. My laptop is a PC which is why I made the switch to Android Studio.

Once downloading Android Studio and familiarizing myself on how it works, I once again started creating my Encoder app. I recreated the wireframe and UI of the application in the activity_main.xml file within Android Studio. From there I began to research what built in functions Android Studio had to offer to help aid me during the development process of building my Encoder app. One of the main functions that I researched and looked into was the java.security function. Basically what the java.security function does is "provide the classes and interfaces for the security framework". I planned to use this package because I saw that it had a bunch of classes that could then be used and implemented for generating and storing cryptographic public keys. After further exploring all the different classes that the java.security package had to offer, I incorporated the Key, KeyFactory, KeyPair, KeyPairGenerator, PrivateKey, PublicKey, Signature, interfaces and spec classes. From the interfaces class it was further broken down into the RSAPrivateKey and RSAPublicKey class. The spec class was also further broken down into the PKCS8EncodedKeySpec, and the X509EncodedKeySpec classes.

Firstly I will explain the spec classes. The PKCS8EncodedKeySpec is used to encode the private key and it ties together with the Key, KeyFactory, and X509EncodedKeySpec classes. The X509EncodedKeySpec class is similar to the PKCS8EncodedKeySpec class, except rather than encode the private key, it will encode the public key. This class is also tied together with all the other classes I mentioned previously. The Signature class also provided to be very useful during my development process because it provided the functionality of creating digital signatures to then be used for authentication. It allowed me to verify the data. What proved to be the most difficult part during this process of creating the app was perfecting the RSA program.

Getting all the classes to work and function properly was definitely a challenge at first. A lot of hours were put into testing, and making sure that everything was running smoothly when encrypting the plaintext and storing that data to then be verified and decrypted. Once the RSA program was written, I then had to tie it all together in the MainActivity.java file. My main source of information was the oracle website and the android studio website. It was a lengthy process, but all in all I'm really happy with how my app turned out!