Project Report: Encryption Based SMS

Braeden Goodwin, Emmanuel Kareem, Amir Mujkanovic

April 22, 2020

Introduction/Problem and Background

Within today's technology filled world, there are multiple different methods of cryptology and thousands of reasons for extra security measures. Our goal with this project was to create a more secure method of encrypting Short Message Service(s) (SMS) on an Android operating system. There are a lot of pair to pair encryption softwares that create a secure way to transfer messages between two parties. These softwares protect the users data in the case that their messages are intercepted. With that being said they do not protect against social engineering (the art of manipulating information from a person without the use of technology). Our main focus for this project was to handle this problem of shoulder surfing. Which is the term used to describe a method used to obtain passwords and other confidential data by looking over the shoulders of an unsuspecting victim.

Implementation Detail:

The implementation process consisted of many different parts that would need to be brought together seamlessly. The different sections consisted of Database design, Authentication, Front-end, Public key encryption, Back-end.

For this assignment we used Firebass to assist us. Firebase is a cloud based mobile and web application software that assists in the development of applications on different platforms.

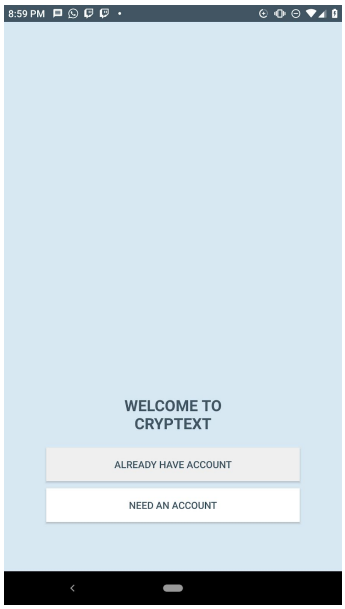
Firebase allows the developers to allocate more time and effort to the core part of the program while it handles things like authentications and storage within the databases.

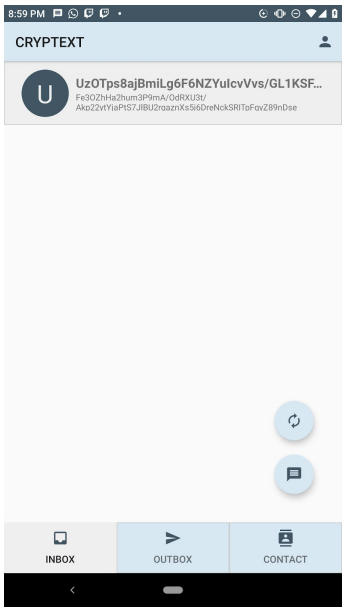
Before the user is able to use the application, they must create an alias using an email address and a password. The user will be able to log into the application using the given email address and password. Once the information is validated, they will then be assigned a public and a private key. These will be unique to the user themselves.

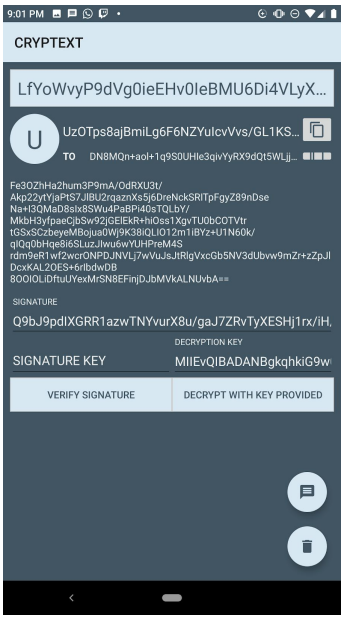
The user's database is divided into two sections, there is the public database where any user can gain access to other users' public keys, emails, and aliases. There is also a private database section which can only be accessed by the user which owns the account. The information that the private database includes are things such as your email, UID, alias, contact list, password, and inbox/outbox.

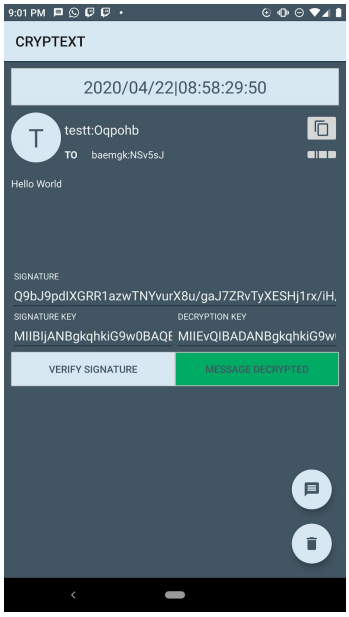
For the public key encryption we used the java.security library to generate key pairs. The keys are encoded from byte arrays to strings. Using the java.security library we are able to encrypt and decrypt strings. We are also able to sign using the public key and verify signatures using a private key that paired to the public key.

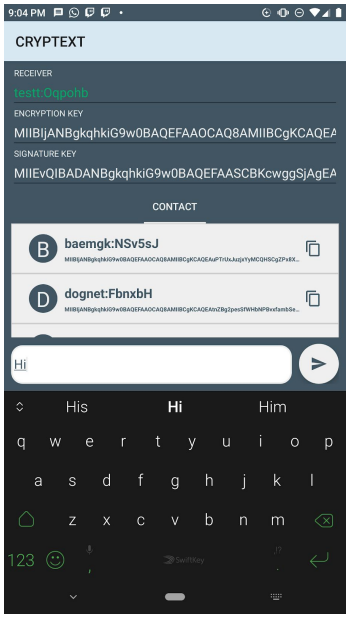
For our front-end development we used Android Studio to design the application and include the functionality of our back-end development. The application has a clean interface that is easy to navigate for new users.


WELCOME TO
CRYPTEXT
ALREADY HAVE ACCOUNT
NEED AN ACCOUNT

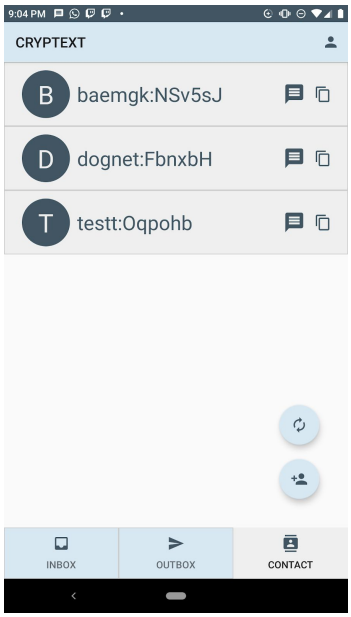

CRYPTEXT
UzOTps8ajBmiLg6F6NZYuIcvVvs/GL1KSF...
INBOX
OUTBOX
CONTACT

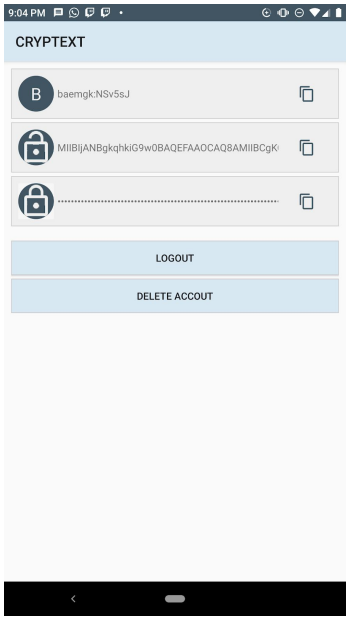

CRYPTEXT
LfYoWvyP9dVg0ieEHv0IeBMU6Di4VLyX...
UzOTps8ajBmiLg6F6NZYuIcvVvs/GL1KS...
TO
SIGNATURE
Q9bJ9pdlXGRR1azwTNYvurX8u/gaJ7ZRvTyXESHj1rx/iH,
SIGNATURE KEY
DECRYPTION KEY
VERIFY SIGNATURE
DECRYPT WITH KEY PROVIDED


CRYPTEXT
2020/04/22|08:58:29:50
testt:Oqpohb
TO
baemgk:NSv5sJ
Hello World
SIGNATURE
Q9bJ9pdlXGRR1azwTNYvurX8u/gaJ7ZRvTyXESHj1rx/iH,
SIGNATURE KEY
DECRYPTION KEY
VERIFY SIGNATURE
MESSAGE DECRYPTED


CRYPTEXT
RECEIVER
testt:Oqpohb
ENCRYPTION KEY
SIGNATURE KEY
CONTACT
baemgk:NSv5sJ
dognet:FbnxbH
Hi
His
Hi
Him


CRYPTEXT
baemgk:NSv5sJ
testt:Oqpohb
INBOX
OUTBOX
CONTACT


CRYPTEXT
baemgk:NSv5sJ
dognet:FbnxbH
testt:Oqpohb
INBOX
OUTBOX
CONTACT


CRYPTEXT
baemgk:NSv5sJ
LOGOUT
DELETE ACCOUT

Testing and results:

We first started testing

Goals	Results
Encrypted user data	User database encrypted on creation of a new user account. The user's public key is used to encrypt their email, alias, and UID. When the users need to access their information the application automatically decrypts it with their private key.
Sending messages between users	Users are able to send and receive messages. When sending a message users will have to input: an alias(the receiver of the message), an encryption key(a key for the message to be encrypted with) and the message itself. When a message is received the user needs to input a decryption key to decrypt the message. This message will usually be the receiver's private key but is not guaranteed. If the decryption key is the right key pair to the encryption key the message is revealed.
App doesn’t crash	Coding standards were followed to make debugging and updating the application straightforward. App crashes have been fixed for the most part and the app runs smoothly.
Message protected if intercepted	If a message is somehow intercepted the intruder will only see the encrypted message and will not be able to decipher it.
Prevents shoulder surfing	Shoulder surfing is a thing of the past because someone with direct access to your phone won't be able see your messages. The only way they might be able to see your message is if they have access to a password since decryption depends on having the right decryption key and the user's password.

Summary/Future Work

To summarize this project we as a group enjoyed our time working on it. It was a challenging experience using methods that we have not previously used in other classes. In the beginning stages, it seemed like coming up with an idea for the project was half the battle. One of the more challenging tasks was retrieving data from the Firebase storage to the front-end. This task was ultimately solved but gave us more trouble than we have anticipated. For future work we do not have anything planned in the near future but we would like to revisit this project and possibly publish it to the Android app store.

References

https://nevonprojects.com/android-based-encrypted-sms-system/

https://firebase.google.com/docs/android/setup

https://developer.android.com/docs