Google Cloud Speech API Based Encrypted Notes

Harsha Kota

Abstract:

This project is the construction of a mobile application for Android that is safe and secure to store sensitive private information such as login credentials and credit card information by implementing AES (Advanced Encryption Standard) to encrypt and decrypt data and additional features such as voice transcription powered by Google's Cloud Speech API for ease of use.

Introduction:

Motivated by the goal of constructing a secure encrypted notes/memo's application that can save any text data such as usernames, passwords, credit card information and all sorts of other login credentials and private information and provide access to decrypted information only after fingerprint authentication so as to protect information from unauthorized access, I have created the application "**Notes**".

This application is fairly secure due to its implementation of Advanced Encryption Standard algorithm to save information only in their encrypted form on a secure local database on the user's mobile.
Encrypting data such as this provides an extra layer of protection against exploitation of stolen data.

Its features are as follows;

- Save notes by typing them in manually
- Save notes through voice dictation powered by Google Cloud Speech API
- All notes are Encrypted with 256-bit AES encryption
- All notes are saved encrypted in an internal database on the phone
- Decrypt notes using fingerprint authentication
- Quickly encrypt notes using shake detection
- A Simple User Interface

Background and Related Work:

Inspired by an idea of an application that displays encrypted data on initial startup and switches to decrypted data after fingerprint authentication and enables a user to shake the device to quickly to turn the application display state to its encrypted state to hide the data from prying eyes.

Existing System:

There are several applications on the Google's Android Play Store that provide a way to save notes/memo's, but none have been found that provide voice transcription using Google's Cloud Speech API. None of them show data in their encrypted state; most of the authentication is done beforehand on the lock screen or before the application is accessed, while these provide enough security, displaying an encrypted message has a certain appeal to it.

Proposed System:

This project builds upon the idea of having such an application that uses 256-bit Advanced Encryption Standard algorithm to encrypt user notes as soon as they are created and stores only the encrypted message in a local SQLite database on the user's mobile.
The key used for encryption will be set during initial launch of the application and is used for subsequent operations of encrypting and decrypting data.

It also provides Speech-To-Text translation when a user wants to enter new notes, this can be toggled from within the new notes screen and also allows simultaneous use of software keyboard and dictation to enter the notes.

On the homepage that displays all the notes, there are options to encrypt and decrypt. Encryption can be done in two ways, clicking the encryption button or shaking the device. Decryption, on the other hand is carried out only after successful fingerprint authentication.

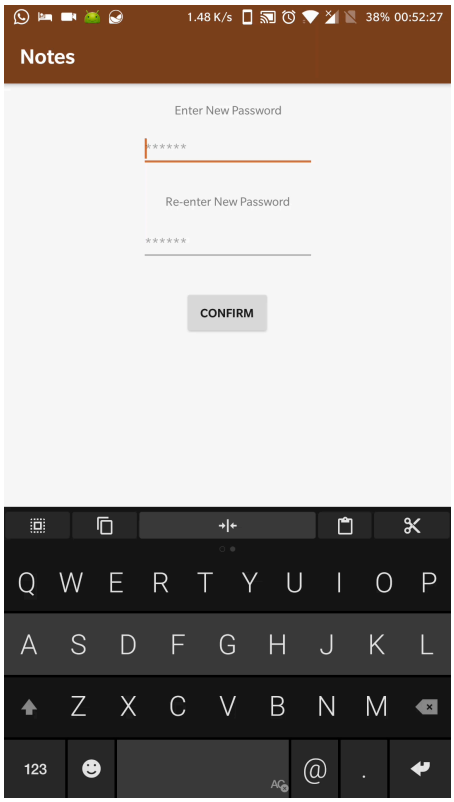
This provides an overall secure application, that's fairly secure to store any type text information.

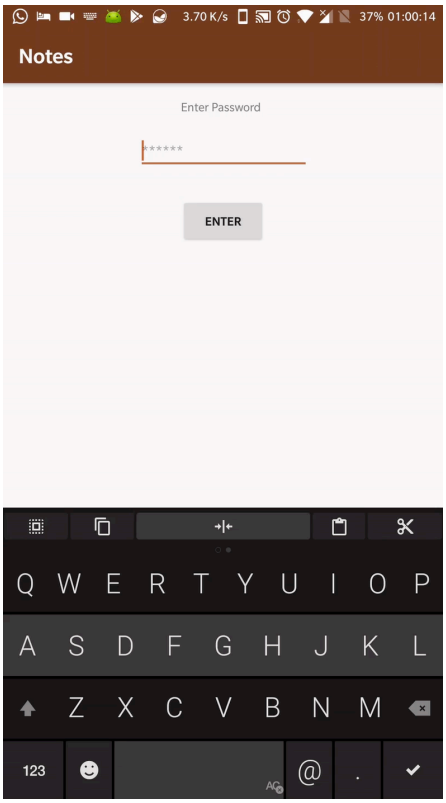
Methodology:

This application is built using Android Studio, Version 3.1 on a MacBook Pro running macOS High Sierra, Version 10.13.4.
It is coded in Java programming language. All libraries used within the application for display and functionality come preinstalled, e.g.

- For UI, Android Studio provides XML to structure the elements in the UI.
- For the implementation of the Advanced Encryption Standard algorithm, Java has an inbuilt crypto library that helps implement any cryptographic algorithm for the application.
- For Shake Detection and Fingerprint Authentication, Android Studio allows for specifying specific Permission Request statements, that upon launch of the application will trigger a request dialog box, that requests for the access on the mobile.

When the application is first launched, the user is greeted with a screen to set up a password for the application. This password can be a text phrase or a numerical password. It is used as a way to enter into the application and also used as a key to encrypt and decrypt all the notes.

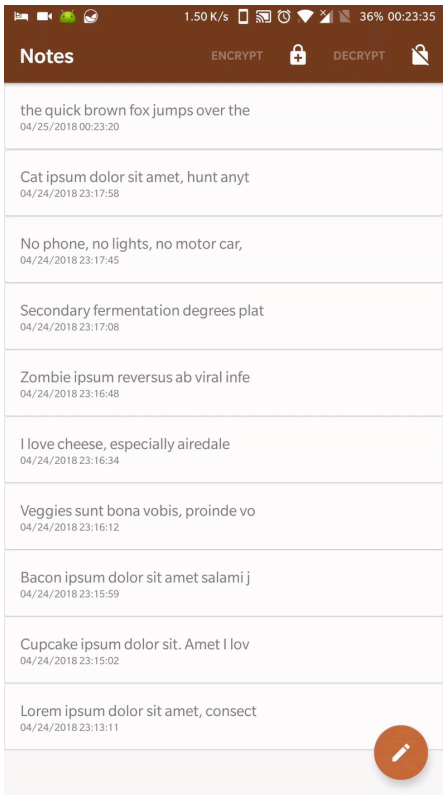


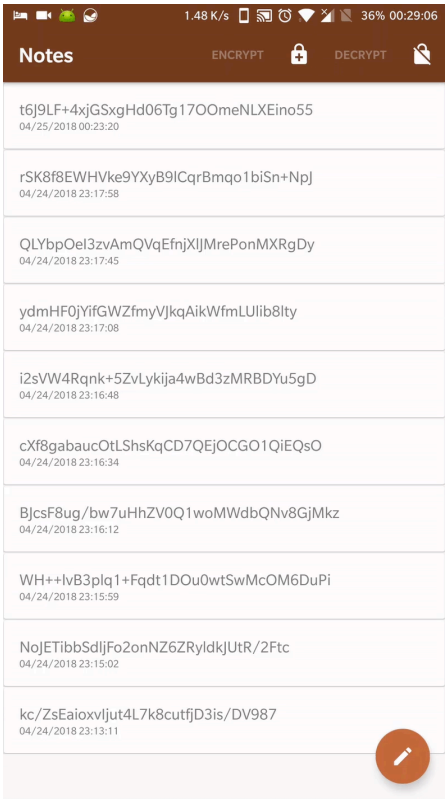


After setting up the password, the user is displayed with a list of previously saved notes, if there are none an Android Toast Notification is displayed on the bottom mentioning “there are no saved notes”. This screen provides two buttons for encryption and decryption of notes if any present, and also a button to add new notes which is active only when the state of the application is in its decrypted state, preventing unauthorized users from making any new notes.

The encryption button, when pressed instantly fetches all the encrypted notes from the database and replaces the list with it, and changes the state of the application to encrypted. This operation is also performed when the mobile detects a violent shake alerting the application to switch all the notes to their encrypted state.

The decryption button, when pressed opens up a dialog box and starts the fingerprint authentication process and waits 3 seconds for a registered fingerprint to be recognized on the fingerprint hardware of the mobile before it returns back to the notes list. On successful recognition of a fingerprint, the application is switched to a decrypted state and all the notes are displayed in their decrypted form. On unsuccessful authentication, the application goes back to the list view of encrypted notes and shows an Android Toast Notification at the bottom indicating “Unsuccessful Decryption”.

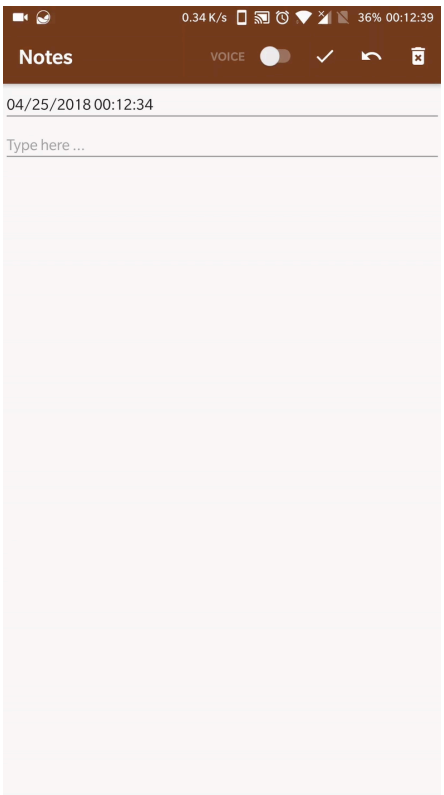




The user can enter new notes using the “+” icon on the home screen of the application. When the application is in the decrypted state, it launches a new screen where the current date and time are pre-populated and a text area is presented to the user to enter data. And when the application is in the encrypted state, it shows an Android Toast Notification to Decrypt before making new notes

On the New Note screen, there are several buttons to Save, Undo, Delete and Voice. The Save, Delete are fairly common and the Undo button shows a dialog box when clicked asking the user if he chooses to discard the data entered and returns to the list view.

When the Voice button is clicked, a request permission pop up is presented to the user to allow the application to use the microphone on the mobile, if this permission is not previously allowed. Once the permission is granted, clicking on the Voice button will start the Google Speech API listener. It listens to and provides partial results in the date, time area of the screen and when the speech ends, it puts the final result into the text area and brings back the date and time.



Experiments:

- Tested the display order of the notes on the main page, which must show notes in descending order of creation/amendment. This is facilitated by an SQLite query with an order by field set to descending. Since each note is associated with a timestamp, its retrieved in such order.
  When new notes are added, or edited, upon returning to the list view, the query is run again to fetch all new notes and old notes and made sure they are displayed in the right order.
- Display voice transcription in real time, replacing the date & time for the duration of speech translation and on the final result from the API the transcription text is added to the body of the note and the date & time are returned to their original state.
- Simultaneous use of keyboard and voice transcription to enter data without any conflicts.

Discussion and/or Analysis:

This application while providing many security features can further be extended with other features such as;

- To provide an alternative to voice transcription using Google's Cloud Speech API, Android's on-board Speech-To-Text API (Recognizer Intent) can be taken into consideration. While Android's onboard implementation doesn't provide powerful speech recognition when compared to the cloud-based service, weighing in on the pros and cons, it poses as a potential additional feature that will help user's when the device is not connected to the internet.
  Pros:
  - It is available even when the application is offline
  - It is free

  Cons:
  - Need to pass local language to convert speech to text.
  - Only works with Android Devices
  - Not all devices support offline speech input.

- The password used to enter the application and Encrypt/Decrypt data is currently being stored in SharedPreferences on the mobile. While SharedPreferences is a secure location that cannot be accessed by other applications, we lose the ability to recover the password, if forgotten. A possible solution can be a cloud based

service that saves notes and only a hashed version of the user's password alongside the notes. This introduces a potential requirement to have a constant internet connection.

- Since, not all versions of Android support fingerprint API, nor all Android devices have fingerprint hardware, we can provide a backup method for decryption. This can be achieved by implementing the lock screen authenticating mechanism, which also eliminates the requirement to remember another pass phrase or pin code.
- Furthermore, Since Google's Speech API is an Enterprise paid product, it's logical to replace it with an Open Source alternative such as Mozilla's Open Source Speech Recognition Model: DeepSpeech.

Conclusion:

Encryption provides an extra layer of protection when saving usernames, passwords, credit card information and all sorts of other login credentials and private information on mobile phones.
In the future, this application can be implemented on various other platforms, including its Google Cloud API feature, as it is not platform dependent or any other alternative that's implemented. Text notes can be further extended to include images, videos to cater to bigger audiences.

References:

1. Ekta Agrawal, Dr. Parashu Ram Pal. (2017) A More Effective Approach Securing Text Data Based On Private Key Cryptography. In: International Journal on Recent and Innovation Trends in Computing and Communication, vol. 5, issue. 3.
2. Suchita Tayde*. (2015) File Encryption, Decryption Using AES Algorithm in Android Phone. In: International Journal of Advanced Research in Computer Science and Software Engineering, vol. 5, issue. 5.
3. Veton Këpuska, Gamal Bohouta. (2017) Comparing Speech Recognition Systems (Microsoft API, Google API And CMU Sphinx). In: Journal of Engineering Research and Application, vol. 7, issue. 3, pp. 20-24
4. https://cloud.google.com/speech/docs/: "Google Cloud Speech API Documentation"