Sound Based Two Factor Authentication

Rahul Sanjay Avvaru
1215133127
rsavvaru@asu.edu

Bhanu Preeti Anand
1217122335
banand2@asu.edu

Maneesha Poluri
1216900399
mpoluri@asu.edu

Harsha Vardhan Kaki
1216885761
hkaki@asu.edu

***Abstract*—Currently there are many two factor authentication mechanisms which commonly receive an OTP (One Time Password) when the user wants to login manually. There are other ways in which the user need not wait for the OTP. Apps such as Google Authenticator provide a solution but the user still has to enter the OTP manually to login. There are other applications like Duo, which takes an extra step in addressing this issue where the user gets a notification with an action button, which the user should press for the two factor authentication to be successful. But all these applications still involve user interaction with the mobile and is inconvenience to the user. In this project, we solve this issue by using a novel sound based authentication approach, which doesn't require user interaction by leveraging to the condition that both the devices are in the same environment and produce the similar sound signatures when recorded.**

***Index Terms*—Multi-factor based authentication, Two-factor based authentication, Ambient sound, User Experience**

I. Introduction

In security, authentication is defined as process of verifying an identity. And in today's world,the authentication problem is huge. According to statistics, in 2016 more than 3 billion password were stolen. The overall cost of cyber crime will sum up to 6 billion dollars by 2021. With occurrence of every such instance, the basic purpose of authentication is failed. Authentication dates back to 6,000 years with the lock and key system. But surprisingly, it is becoming worse. The main reason behind it is "shared secret", in other words sensitive data like passwords, pin codes, credit card numbers, social security numbers are everywhere. There are various problems with shared secrets like it is not unique. But we have an alternative solution to shared secrets, it is better authentication [1]. This is can be achieved by multi-factor authentication. Multi-factor authentication is defined as an authentication method in which a computer user is granted access only after successfully presenting two or more pieces of evidence (or factors) to an authentication mechanism: knowledge (something the user and only the user knows), possession (something the user and only the user has), and inherence (something the user and only the user is).Two-factor authentication (also known as 2FA) is a type, or subset, of multi-factor authentication. It is a method of confirming users' claimed identities by using a combination of two different factors: 1) something they know, 2) something they have, or 3) something they are. Advance research in two factor authentication for mobile and web applications consider different second factor, which provides user friendly experience. Accuracy improvements in measurement by microphones lead to two-factor authentication based on ambient sound [4].Here, proximity of ambient sound is determined by recording the noise of user location from mobile phone and a laptop present in the same room. This is an effective two factor based authentication.

II. Project Setup

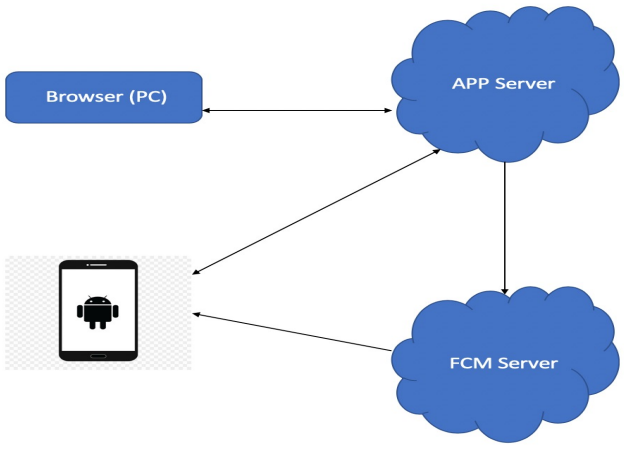

Fig. 1: Architecture.jpeg

The architecture consists of 4 components. The components are described below:

A. Android device

- Model and Make: Samsung Galaxy S9 plus
- OS: Android 9.0 (Pie)

B. Cloud Server

For testing purposes, we used local server.

- Make and Model: Mac Book Pro
- OS: Mac OS Catalina
- Processor: 2.6 GHz 6-Core Intel Core i7
- RAM: 16 GB 2400 MHz DDR4

C. PC

Device on which the user will login to the website.

D. Firebase Cloud Messaging Server (FCM)

Android device and Browser (Device on which the user is trying to login to the web page) will be communicating with the application server and will not be communicating directly. Application server will communicate with FCM Server and FCM Server will send the necessary push messages to the Android device.

III. Implementation

The implementation is divided into the following four major tasks:

1. Web application

This is the web application that the user is trying to access. The user is required to enter a username and a password to login to the system.

The below screenshot shows the login page that the user is trying to access:

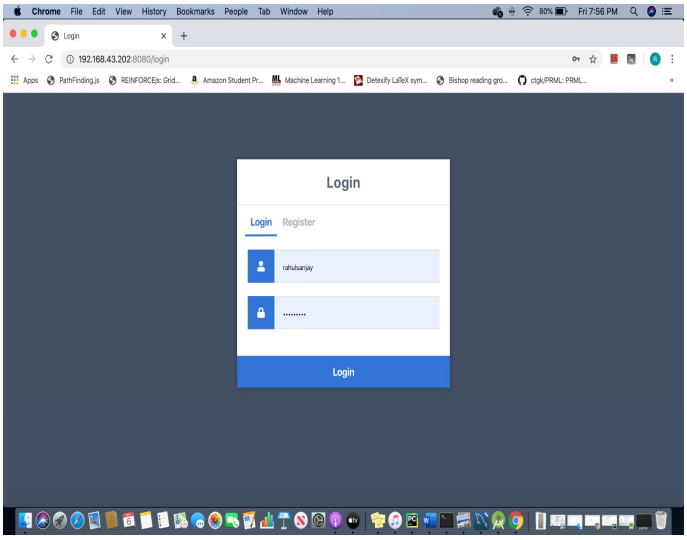

Fig. 2: Login Page

Once the user clicks the login button, a login request will be sent to the application server and if the user credentials are valid and in the database server, the user is redirected to the performing two factor authentication page. If the username or password is invalid, the user is redirected to the login page.

The below screenshot shows the two factor authentication waiting page:

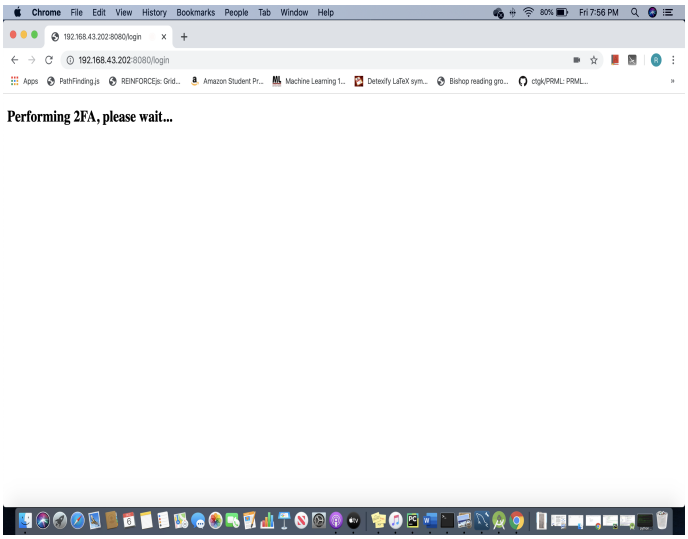

Fig. 3: Waiting for the two factor authentication to complete

If the two factor authentication is successful, the user is redirected to the home page.

The below screenshot shows the homepage of the web application:

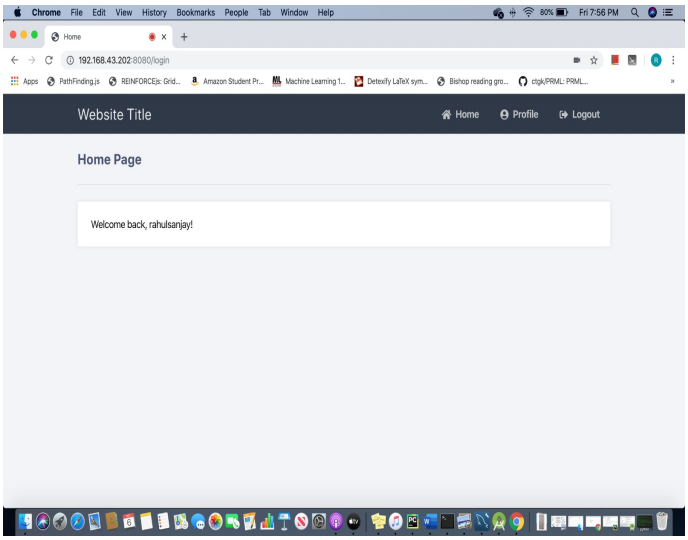

Fig. 4: Home Page

If the two factor authentication fails, then the user is redirected to the login page with an error message.

The below screenshot shows the login page with the error message:

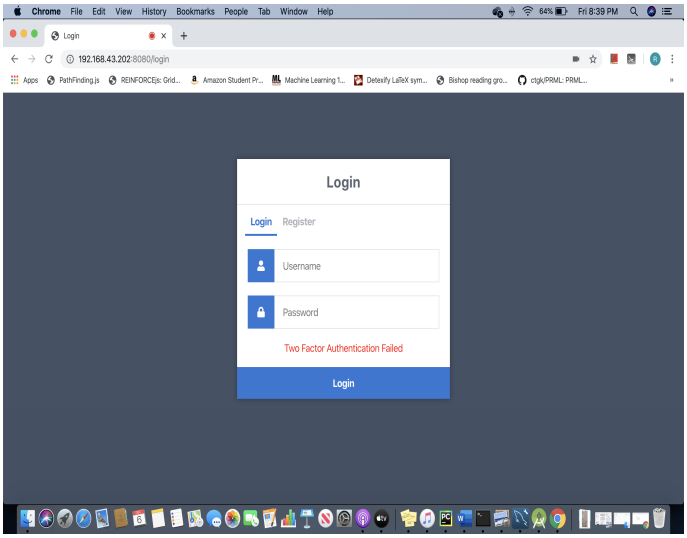

Fig. 5: Two Factor Authentication Failed

2. Android application

The Android application consists of two services.

- MyFCMService.java
- SoundRecorderService.java

FCM service architecture is shown in figure :

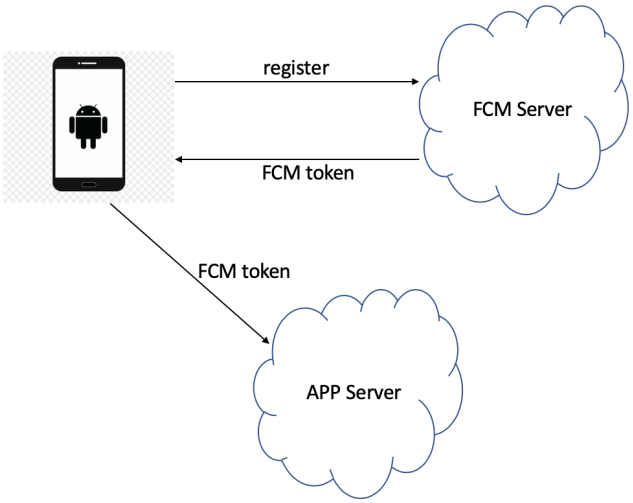

Fig. 6: Firebase cloud messaging server architecture

FCMService receives two types of push messages from the application server. ACTION_RECORD which instructs the android application to start recording the sound for 5 seconds. ACTION_SEND_FILE which instructs the android application to send the recorded file to the server.

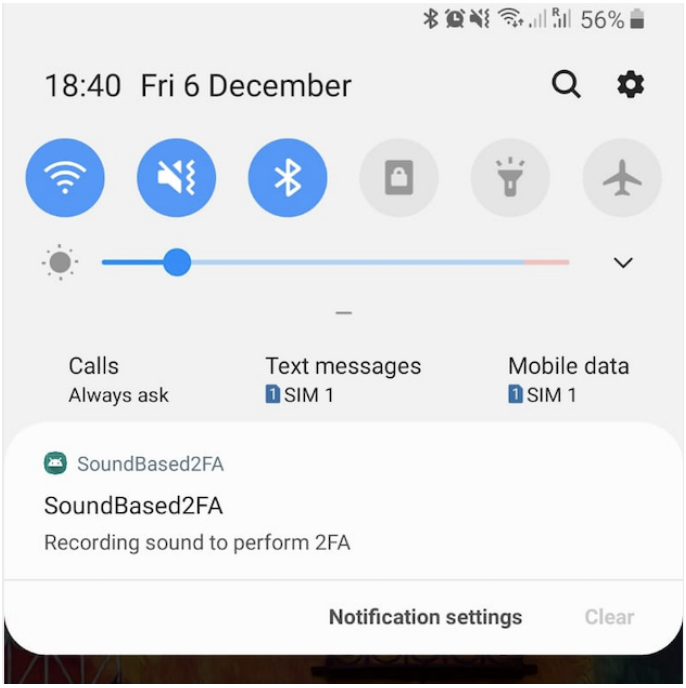

Fig. 7: Notification in Android device

When the android application receives the ACTION_RECORD, it starts the SoundRecorderService as a foreground service. On devices running Android 9 (API level 28) or higher, apps running in the background cannot access the microphone. Therefore, our app can record audio only when it's in the foreground or when we include an instance of MediaRecorder in a foreground service. As we don't need any user intervention, we record the sound from a foreground service which displays a status bar notification during recording.

The above figure-7 shows the notification that is displayed when the android device is recording the sound:

3. Application Server

This is the central component of our system which communicates with the android device, PC and also the FCM Server to send the push messages to the server. For our testing purpose, we had setup a local server.

The below screenshot shows the running local server(figure-8):

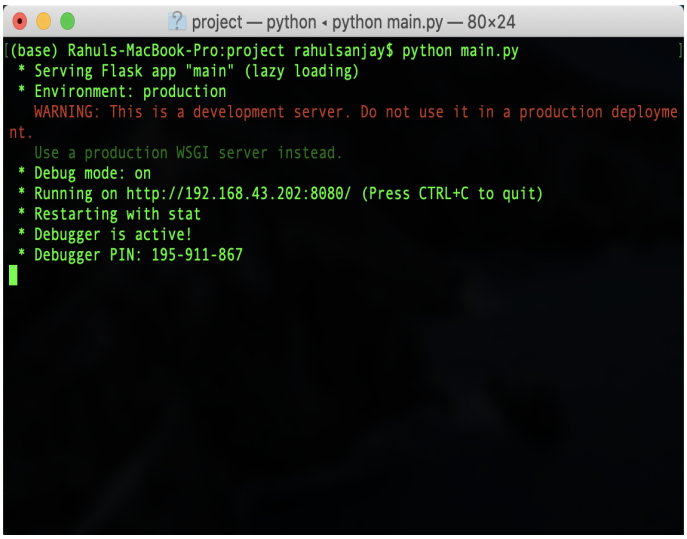

Fig. 8: Local server

When the server receives a login request from the user, it instructs both the PC and the Android device to start recording. Figure-9 shows the same.

After receiving the start recording request, both the PC and the android device will record the sound for 5 seconds. Once the recording is done, the PC sends the recorded file to the server. Once the server receives the file from the PC, it sends a request to the android device to send the recorded file. The android device then sends the recorded file to the server. The server saves both the files in its root directory and then uses the correlation logic to calculate the similarity score between the two audio files.

If the similarity score is greater than a threshold, we allow the user to login. In our case, we assigned 0.5 to threshold

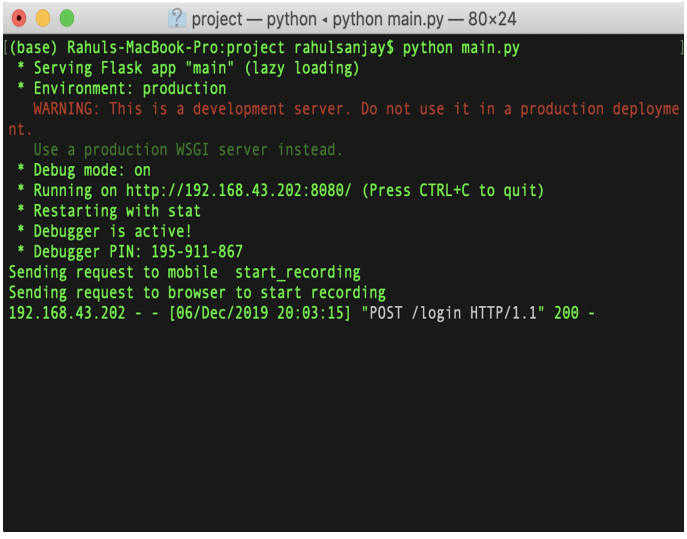

Fig. 9: After receiving the login request

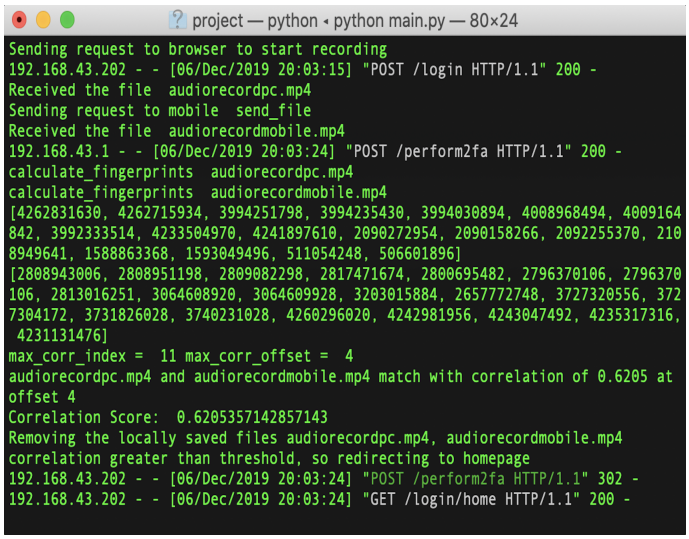

Fig. 10: Successful login scenario

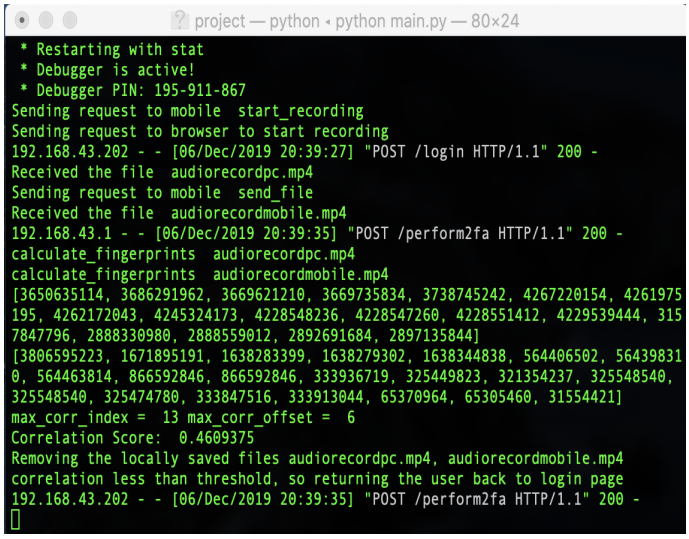

Fig. 11: Two factor authentication failed

Figure-10 shows the successful scenario where the threshold is greater than 0.5.
When the threshold is less than 0.5, the two factor authentication will fail. The failure scenario is shown in Figure-11.

4. Correlation logic

The correlation logic uses a command line tool called fpcalc to extract the acoustic fingerprints from both the audio files and calculates the similarity score which is between 0 and 1. This file is included in the zip folder. One important thing to notice here is that there might be delay between the PC and android device when they start recording the sound. This is due to the communication time between the server and the android device. This is handled in the correlation logic by comparing the extracted acoustic fingerprints with offsets spanning over a range.

IV. Completion Of Tasks

SL.NO	TASK	ASSIGNEE
1	Setting up GitHub for the project	Maneesha
2	Setting up a web server to store login credentials	Harsha Vardhan
3	Creating a web service to handle the incoming requests	Bhanu Preeti
4	Handling the login request	Rahul
5	Sending request to PC to start recording	Maneesha
6	Sending request to mobile to start recording	Harsha Vardhan
7	Creating a login page	Bhanu Preeti
8	Receiving response from server and start recording	Rahul
9	Creating an android app to receive the push from server and start recording	Maneesha
10	Sending the recorded sound data to the server from the app	Harsha Vardhan
11	Sending the recorded sound data to the server from PC	Bhanu Preeti
12	Code to compare the audio	Rahul
13	Code to compare the audio	Maneesha
14	Code to compare the audio	Harsha Vardhan
15	Code to compare the audio	Bhanu Preeti
16	Returning successful or failure authentication to the user	Rahul
17	Successfully authenticating the user or showing failure with proper UI in the webpage	Maneesha
18	Generating random key from pc	Harsha Vardhan
19	Generating random key from mobile	Rahul
20	Mobile App UI	Bhanu Preeti

Fig. 12: Tasks Completed

V. Conclusion

The project is based on two factor authentication, in which a web application is authenticated based on sound as the second factor. Currently, the second factor authentication provides bad user experience as it needs their approval. Ambient sound is recorded by browser and mobile application for 5 seconds and sent to server to perform correlation on the generated fingerprints of audio files. The similarity algorithm decides to complete the second factor authentication based on the threshold factor. If similar sound is not detected then the authentication fails, thereby restricting the user from moving ahead. This methodology of second factor authentication successfully provides user friendly experience and a robust security.

VI. Acknowledgment

We would like to thank our Professor, Dr.Ayan Banerjee for guiding us throughout the project with various inputs and providing help for our queries. We would also like to thank TA, Ms.Sameena Hossain for her inputs.

References

[1] https://medium.com/tokenring/the-worlds-authentication-problem-accfda2a30e6
[2] https://github.com/QuickLyric/fpcalc-android
[3] https://medium.com/@shivama205/audio-signals-comparison-23e431ed2207
[4] https://en.wikipedia.org/wiki/Multi-factor_authentication