Zap! : Three alternative approaches to traditional text-based authentication Design, development and evaluation

Yujia Chen
School of Computer Science
North Carolina State University
Raleigh, NC 27695
Email: ychen71@ncsu.edu

Kaustubh Gondhalekar
School of Computer Science
North Carolina State University
Raleigh, NC 27695
Email: kgondha@ncsu.edu

Siddharth Sharma
School of Computer Science
North Carolina State University
Raleigh, NC 27695
Email: ssharm24@ncsu.edu

***Abstract*—Authentication is everywhere around us, the continuously grown Internet services have given us more accounts than we can manage. Most of the services need text-based passwords, since different services requires different policy to create the password, which may lead to different passwords for users to remember. Our project, Zap!, uses an Android application to explorer three different ways for authentication: fingerprint, voice recognition, and magic link. We then continue to conduct user evaluation using questionnaire and focus group, collecting and comparing data and feedback from the evaluation.**

***Keywords*—Authentication, biometrics, fingerprint, voice recognition, password-less logins, graphical password**

I. Introduction

Increasing number of Internet services like social networking, video-streaming, e-commerce shopping, gaming etc. require the user to remember myriad passwords. Every service demands a different policy of creating a password for their service which the users need to satisfy. On top of that, no service takes into account the cognitive pressure levied on the users from concocting unique and weird passwords and then remembering them. Password management is deemed entirely the user's responsibility [2].

From the literature review we did on this subject in our February report, we gathered information from materials to conclude the user-login problems: Text based passwords are painful for users to remember; Most of them are vulnerable to dictionary attacks; They are not easy to use. In order to get a better perspective on this problem, we conducted a user-survey in the February report, and we are glad to find out that most of our participants agree with us. Based on those results from our February report, we finalized three approaches we think are the most promising to substitute the traditional text-based password: fingerprint authentication, voice-DNA authentication and using "Magic link" via email for authentication.

We decide to create an Android application to explore those three approaches, since Java is the most common programming language among us three members in the group, and also Android has a neat platform providing enough APIs for us to create a quick prototype for user evaluation.

To provide a better simulation of user's login process as well as eliminate other factors that may affect user's login process, we also built a webservice for user authentication. The server acts as a centralized hub to handle all the real authentication process. Underneath the three different implementations, we still use password to authenticate the user. This password is chosen by the user on account creation, regardless of which method he uses afterwards to login. We chose Ruby on Rails as our Web framework, which has several plug-ins and functionality for authentication. Inside, we have RESTful interfaces created for API calls, such as create user profile, password matching, password update, authorization token verification, magic link.

Throughout this report we will discuss how we implement those three authentication methods and the user evaluation. Then we will use the data and feedback from the user evaluation to conclude what we think is the most promising solution.

II. Synopsis

A. Previous work

The literature presents lots of solutions to this problem. In today's modern era, almost everyone owns a 'smart-phone'. "Mobile devices are rapidly becoming a key computing platform, transforming how people access business and personal information" [6] Making users type out weird symbols and alpha-numeric characters on the small screens of their smartphones takes away all the great usability that a mobile device offers. A recent paper on biometric authentication beautifully states "Biometric authentication is often referred to as the secret weapon of authentication, mainly due to the fact that the password (e.g. a fingerprint) cannot be forgotten"[5]. It seems we have hit two birds in one shot! We now have the portability of a mobile device combined with the usability of a biometric password! The literature points to using biometrics as a solution to the plague that text-based passwords have become. Using biometrics also present us with lots of choices: fingerprints, voice, face-recognition, keystroke-DNA, gait analysis etc. The first biometric we chose (fingerprint) needs no justification in today's world. Apple's TouchID as well as Android's fingerprint authentication have taken the world by the storm. This feature is immensely popular and is a great candidate for our problem. We eliminated out keystroke-DNA

and gait analysis as it wasn't a mature enough in both literature and real-world use. The title of the paper on understanding biometrics on smartphones explains our elimination of facial recognition; the paper is titled 'I Feel Like Im Taking Selfies All Day! Towards Understanding Biometric Authentication on Smartphones'[7] While analysing facial-recognition, the paper mentions "Besides bad lighting conditions leading to bad performance, the participants mentioned social awkwardness as an important external factor that kept them from using the system."[7]

B. Server

For user management, we chose the Devise gem. Devise is a flexible authentication solution for Rails based on Warden, it is Rack based and based on modularity, so we can chose to use what we need and get rid of the rest. The latest version of Devise deprecated the authorization token, but since we need to use the authorization token for magic link, we manually generate an authorization token and assign it to the user. When the user registered with the server, it will assign an authorization token to the user, once the server has authenticate the user using password matching, it will send the authorization token to the user, and the user can keep it as their pass to access services on the server. Once the user send a logout request, the server will assign a new authorization token to the user.

Since we want to eliminate the possibility of user input the wrong email address, so they can use the magic link authentication without any trouble, we implement a verification process, which will send an verification email to the email address after user registration, once the user clicked on the link inside the email, the registration process is complete.

The server is deployed on Heroku, the database which Rails ActiveRecord is accessing is PostgreSQL, which is free to use on Heroku.

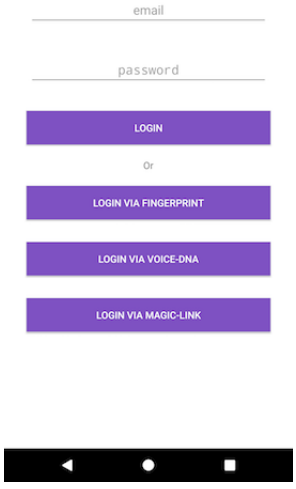

Fig. 1. Login screen of Zap

C. Fingerprint Authentication

This method of authentication requires a fingerprint enabled android device. The user is also required to register his/her fingerprint into the device before using this method. Since android stores its fingerprints inside hardware, we have no way of storing users fingerprints in any form at the backend and therefore have to rely on the methods provided by Android to verify fingerprints.

The user enters his/her email (username) and clicks on the button login via fingerprint. This takes the user on a new page where he/she is prompted to place his/her fingerprint on the sensor. Once the user is verified, a backend API call sends the users email (username) and password to the server and takes him/her to their profile page on successful verification of the credentials.

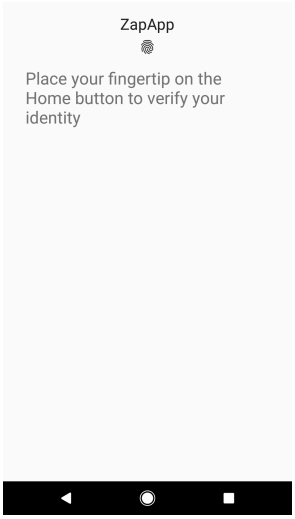

Fig. 2. Fingerprint Authentication

D. Voice-DNA Authentication

This method of authentication verifies the user by his/her voice. We have used Microsoft's Cognitive (Speaker Recognition) API. Authentication via voice requires the user to follow a set of steps to set it up.

Step 1. Profile creation and Enrollment:
Profile creation is done automatically for the user. For the enrollment we require 3 voice samples. These enrollments depend on a phrase which is provided by the Microsoft service, which means the user enrolls and verifies using the same phrase.

Step 2. Verification:
Here the user records the same phrase with his voice to authenticate. The sample first goes through to Microsoft which matches it against the previously provided samples. After it is successfully validated we send a request to our server to login the user.

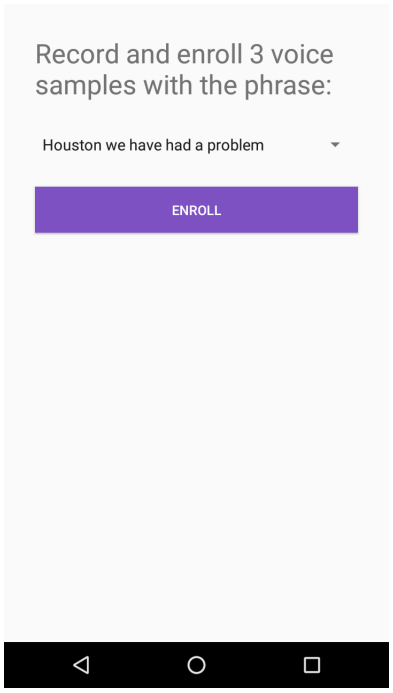

Fig. 3. Voice Authentication

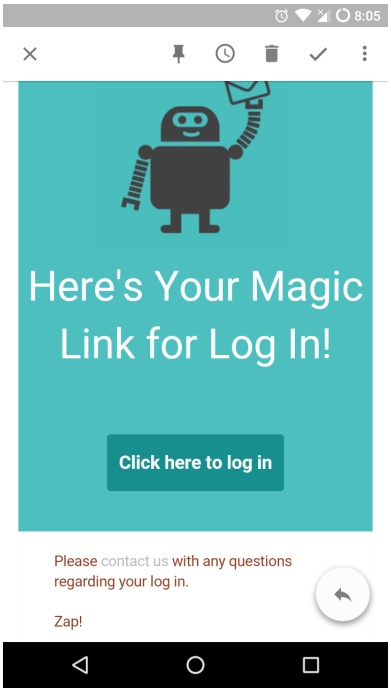

Fig. 4. MagicLink Authentication

E. MagicLink Authentication

A magic link is a unique url, which when clicked signs the user into his account. This link is sent to a users email, which is a users primary identifier and needs to be verified before using this method). This type of authentication is currently in use at platforms like Slack(chat), and Medium(blogging), and the authors were inspired by them. Here, the user only needs access to his email to login. Magic links need to secure such that they shouldnt be guessable, and once a link is used to login, it should not be reusable; and the link should expire after some amount of time.

To login via this method, on the Android side the user enters his/her email and selects 'login via magic-link' button. This event triggers an API call to ZapServer prompting it to send the 'magic-link' on the user's email address. The core functionality of magic-link is a unique authorization token generated by the server, for a user account. When the user clicks on this link, he is directed inside our App where the auth-token is parsed from the link. A subsequent API call is then made to verify that auth-token with the server. On successful verification, the user is logged in.

For the server, when the user send a magic link sign in request to the server, the server will send the authorization token to the users email address in the form of a web link. This is done with Rails ActiveMail service and Gmail, we configure the smtp service for Gmail. Once the user receive the email and click on the link, it will re-open the application on users Android device and parse the authorization token in the link. Then the application will send the authorization token along with their email address to verify the authorization token. If it is successfully verified, then we determine that the user has successfully signed in.

III. User Evaluation

In this section, we will talk about the approach we took to conduct the user evaluation. Since we already have three implementations on user evaluation, we need to get user feedback, observe the user login process and gather data from the process.

A. Evaluation Plan

Since we have an Android application for the three authentication methods, we are using an Android device with fingerprint sensor. All participants will be using the same phone to eliminate other factors that may affect the result. Since we are trying to find a authentication method that may be the replacement for the traditional text-based password method in the future, we have a built-in authentication method using text-based password as the base line.

We are also interested in the enrollment process. Not only should the authentication method be easy to use for the users, but it also needs to be easy to enroll as well. We used a stop watch to time the user's enrollment process for text-based password, fingerprint and voice recognition. Since magic link does not require an enrollment process, we will assume the enrollment took 0 second.

To enroll the text-based password is straight forward, the user registers a new account in the application using email, input his/her password twice (one for confirmation), then click on register to complete the enrollment process. In the login page, the user input his/her username, which is the email is this case, then input the password using the keyboard on the phone screen, click on the login button to complete the login process.

For the fingerprint authentication, the enrollment process is that the user navigates into system settings, and records

his/her fingerprint in the phone via the fingerprint sensor. To login his/her account via fingerprint, on the main screen of the application, the user clicks on ”login using fingerprint” button, a screen will show to guide the user using their fingerprint to login via the fingerprint sensor.

For voice recognition, the enrollment will require the user to choose a sentence from 10 options listed on the screen, then the user will be asked to click on the enroll button to record his/her voice and then sent to the server. The enrollment will require 3 successful takes after which, if the user has already enrolled or just finished the enrollment process, a verify page will show to lead to user to verify his/her voice using the sentence the user chose during the enrollment process. If successfully verified, the user profile page will show, if failed, a waring will appear asking the user to verify again.

For magic link, the user only need to click on ”login via magic link” button, and an email containing the link used for login will be sent to user's email address. After the user click on the link in the email, it will take the user back to the application and complete the login process.

We also prepared a survey for the participants to fill out once they finished using all four authentication methods. The participants will rate their experience with fingerprint authentication, voice recognition authentication and magic link authentication, then rank those methods.

At the end of the user evaluation, we asked the participants for their feedback on our application, how they feel like each authentication methods, and what we can do to improve the experience.

B. *Participants*

The majority of participants in this user evaluation are students from the Software Engineering course in Spring semester, 2017. They are graduate students studying Computer Science at North Carolina State University and know their way around software, they have experience with applications on mobile devices and with user authentication as well. The other participants are graduate students at North Carolina State University, their major varies from Agriculture to Financial Mathematics. They have experience with either iOS or Android devices, some of them never used the fingerprint sensor before, even though most of them are not tech savvy, they have experience with user authentication.

IV. Result

The user evaluation took place at a study room in James B Hunt Library, and device used in the evaluation is Huawei Honor 5X, with a fingerprint sensor placed in the back. The timing result for enrollment and log in process using traditional text-based password authentication methods is shown blown in Figure 5 and Figure 6.

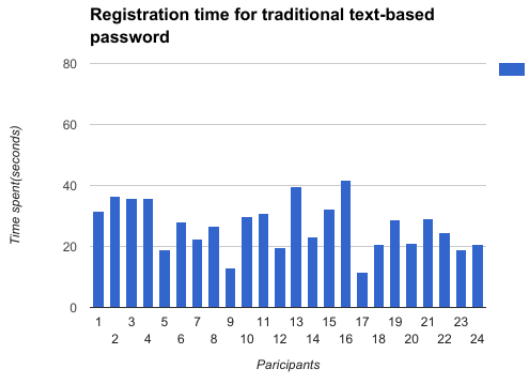

Fig. 5. Enrollment time for text-based password

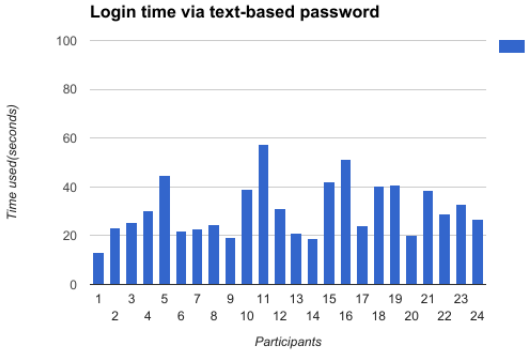

Fig. 6. Login time for text-based password

As we can see from the time cost for users to register their accounts using the traditional text-based password, this process has been seen in various applications and we have the most basic approach, so this does not take too much time to do, as for the log in process, we want to see the worst case scenario, which is the user is not familiar with their account, so we let the user to log in with our given credential, which is an 18- character email address and an 8-character long password, containing 4 lowercase letters, 1 uppercase letter, 1 symbol and 2 numbers. Time varies from the participants, but generally it takes at least 20 seconds for the users to log in. 3 of the participants failed to log into the account in one take.

Figure 7 and Figure 8 shows the enrollment time and log in time for fingerprint authentication. For enrollment, we go into the Android system setting, participant needs to put his/her finger against the fingerprint sensor, lift the finger, and put it back on, and repeat until the process completed and the fingerprint is log into the system. We need to point out that this is a brand new phone, and all of the participants have never used this phone before. We didn't take the time used for the user to get into the system setting, just the fingerprint setting alone. Time varies since participants' experience with fingerprint sensors and Android phones varies, some are very familiar with the process and it went smoothly, some need 2 takes to finish the setup. For log in, the whole process is quick and accurate, all the participant log into their account in 1 take.

Figure 9 and Figure 10 shows time the 24 participants spent

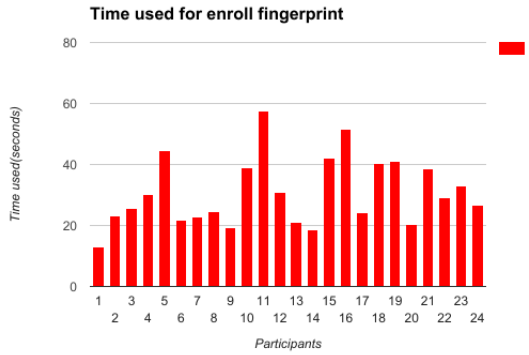

Fig. 7. Enrollment time for fingerprint authentication

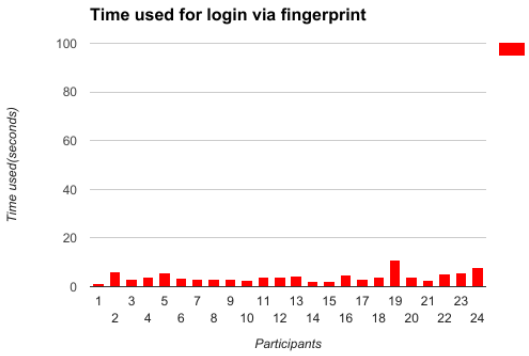

Fig. 8. Login time for fingerprint authentication

to enroll into voice recognition log in and to log into their account using their voice. The process for enrollment is first, the user has to allow the application to gain access to the microphone and storage, since it will require microphone to record the sound and will need storage to store the recording and send that data to web-service. Then the screen will show, let the participant chose from the 10 sentences, the participant has to use the 1 sentence to enroll and log in. The participant will click on the enroll button to begin the enrollment process, he/she will say the sentence, click on the pause button, then click on the send button on the right corner of the screen, then

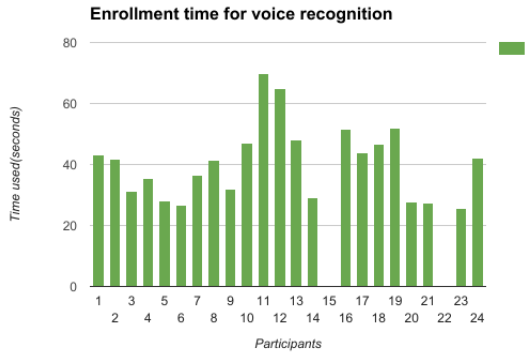

Fig. 9. Enrollment time for voice authentication

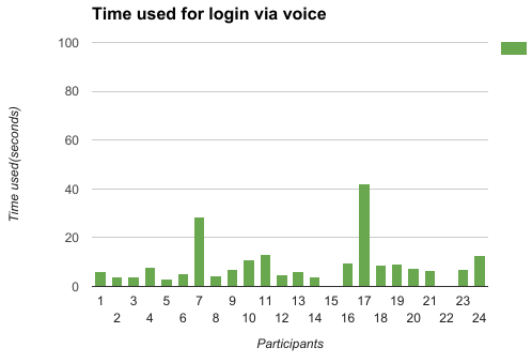

Fig. 10. Enrollment time for voice authentication

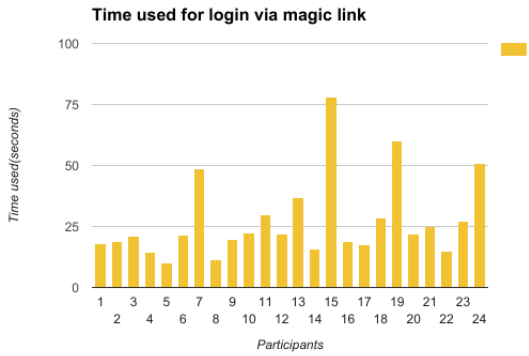

Fig. 11. Login time for magiclink authentication

repeat the process for two more times if the first one succeed. After 3 successfully takes, the enrollment process end and log in process can begin. For log in, the participant click on verify button, record the sentence, click on pause button, and then click on the send button on the right corner of the screen. There are 2 participants fail to finish the voice enroll and log in process, because we are using Microsoft Cognitive API for voice recognition, and the server is on west coast, since we are on east coast the service is not stable, and the 2 participants didn't get the chance to do it since the service went down during their evaluation. All participants take a long time for enrollment since it will require 3 takes to finish the process, for the log in process, 22 participants successfully logged into their account using 1 take.

For magiclink there's no enrollment process, since that is already been included in the traditional text-based enrollment process. As for log in, the user will click on login via magiclink button, and an email will be sent to the participant's email address, the participant click on the link contained inside the email, and it will take the user back to the application on the phone. Time differs from one participant to another, main reason is the email being sent takes sometime to get to the email address, and needs time for the push notification on the phone. The participant needs to go into the email client to check the email, which also needs time.

Aside from those data, we also used a user evaluation form prepared for the participants to fill out. The survey is

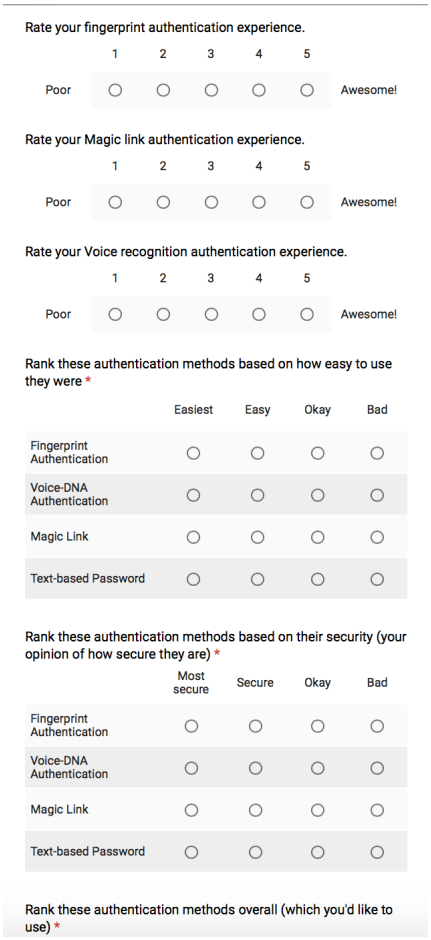

Fig. 12. User evaluation survey

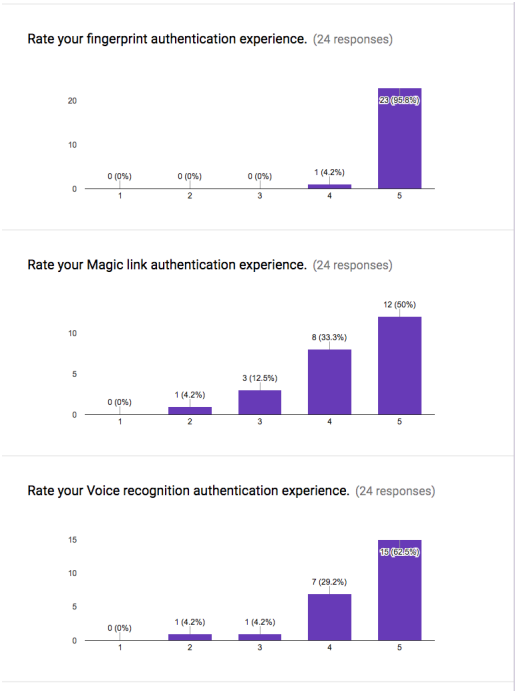

Fig. 13. User evaluation survey

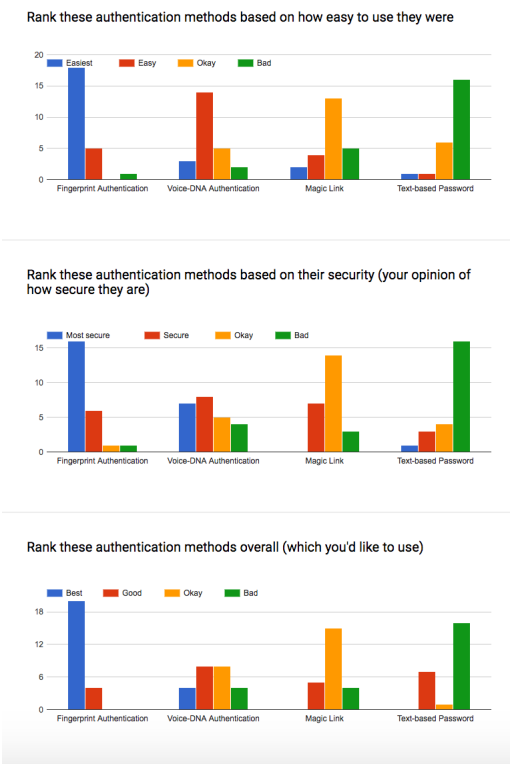

Fig. 14. User evaluation survey

shown in Figure 12 and the result is shown in Figure 13 and 14. The participants think our three implementations are user friendly and easy to use, we also let the participants rank the 4 authentication methods, text-based password, fingerprint, voice recognition and magiclink, by how easy they are to use, how the participants think the level of security those methods have, and how they like those methods overall.

We can see that on all three fronts viz. ease of use, security and overall comparison the fingerprint-authentication fares the best and the text-based password authentication fares the worst.

V. Discussion

A. Observation and Limitation

Traditional Text Based Authentication

This method is the tried and tested, mundane one which no one had any problems 'understanding'. Though most of the people are quite adept at using the small scrawny keys on the small touch screen keyboard; some faced problems logging in and managed it correctly on the second or third time. The typing of email and password took most of the time executing this approach.

Magic Link Authentication

Some users were aware of the magic link method of authentication, which they remembered as the one used by Slack messenger. Participants found the magic link to be a bit cumbersome to use on a mobile device. This was because of the kind of two step process to login viz. Press the 'login via magic link' button, then open your e-mail and click on the link sent to authenticate. Also most said that since this involved hopping through two applications, they would be a bit wary of using it primarily instead of the normal text-based authentication. The usage of magic link also was contingent on prompt delivery of email which sometimes caused some delay in the authentication process.
Thus using the magic-link on a mobile device could be termed as its limitation according to the users we interviewed. The magic link authentication could be much more effective if used on a desktop environment.

Fingerprint Authentication

This method takes on an average 25 seconds to enroll albeit its the fastest one to login the user once this is done. Almost all of the users 'loved' this method of authentication since it required virtually no effort on the part of the user. Most of our participants were well versed with fingerprint authentication, since they already were using the fingerprint sensor on their mobiles. For some participants (mainly those who weren't well versed with the use of the fingerprint sensor) they had some snags while doing the fingerprint registration process but those were pretty minor and they had no problems after they got the hang of using it.
The only limitation to this method of authentication is procuring a fingerprint enabled device. But since such devices are only growing more and more common with time, availability can be easily taken care of.

Voice Authentication

This method was rated as the 'cool' one for authenticating by most of our participants. The feeling of authenticating by your voice gave probably gave them kind of a futuristic feeling albeit, this method was rated as the least practical. Most users stated that they would feel weird speaking into a phone in a public place to login using weird phrases (e.g huston we have a problem)! Also participants cited some obvious problems with this method viz. potential problems logging in in a noisy room/loud public places. Some also cited security vulnerabilites like using a recorded voice sample to login.
Although this method seems quite futuristic from the outside, it has a number of limitations as mentioned above. It has security concerns and requires a lot of conditions e.g a reasonably quite setting, a normal voice (if you are down with cough/cold you will be locked out of your own device!) etc. Thus based on the current state of the systems, the practicality of this method of authentication is certainly an issue.

B. Best Feature

Throughout the user-evaluations and surveys there seems to be a unanimous winner: the finger-print authentication. Although had the magic-link authentication been implemented on a desktop environment, we believe our results would have been slightly different. But there is indeed no denying of the fact that fingerprint authentication is the easiest to use on a mobile device. The magic link we believe is more geared towards a desktop environment. The voice authentication lacks a good infrastructure and implementation based on the current systems and will probably be more practical some years down the road.

C. Future Work

The user evaluation method could be made randomized, to get more accurate results. Also our evaluation consisted of recording data of first time usage only. Looking at long term data would give a better idea of effectiveness of these methods. Our delivery platform was mobile, and some methods like voice are more suited towards stationary technology like Alexa or Google Home, usually in a home environment. Similarly magiclink is more geared towards desktop or laptop usage, where switching apps is easy, and fingerprint and voice may not be available.

VI. Conclusion

In this project, we try to find out the best alternative way for authentication. Even though all have disadvantage at some point, for example, voice recognition won't be effective in certain places such as the quiet room in the library, magiclink authentication needs to jump between apps in our implementation, but between all these three methods, we believe fingerprint authentication is the best way. It's the fastest and easiest to use among the three methods.

	usability	privacy	special hardware	offline ability
text	+	-	+	-
fingerprint	+	+	-	+
voice	+	-	-	-
magiclink	+	+	-	-

TABLE I. Trade off table for different authentication methods

But that doesn't mean the other two is completely useless, as we can see in the trade off table, they all shine one way or another in different applications. We look forward to those technologies continue to improve, and we believe there will be a way to substitute traditional text-based password eventually.

REFERENCES

[1] Saranga Komanduri, Richard Shay, Patrick Gage Kelley, Michelle L. Mazurek, Lujo Bauer, Nicolas Christin, Lorrie Faith Cranor, and Serge Egelman. 2011. Of passwords and people: measuring the effect of password-composition policies. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11).

[2] Shirley Gaw and Edward W. Felten. 2006. Password management strategies for online accounts. In Proceedings of the second symposium on Usable privacy and security (SOUPS '06). ACM, New York, NY, USA, 44-55.

[3] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot, and Frank Stajano. 2015. Passwords and the evolution of imperfect authentication. Commun. ACM 58, 7 (June 2015), 78-87.

[4] Beate Grawemeyer, Hilary Johnson, Using and managing multiple passwords: A week to a view, Interacting with Computers, Volume 23, Issue 3, May 2011, Pages 256-267, ISSN 0953-5438, http://dx.doi.org/10.1016/j.intcom.2011.03.007.

[5] Bhagavatula, C., Iacovino, K., Kywe, S. M., Cranor, L. F., and Ur, B. Poster: Usability analysis of biometric authentication systems on mobile phones. SOUPS Poster (2014).

[6] Shari Trewin, Cal Swart, Larry Koved, Jacquelyn Martino, Kapil Singh, and Shay Ben-David. 2012. Biometric authentication on a mobile device: a study of user effort, error and task disruption. In Proceedings of the 28th Annual Computer Security Applications Conference (ACSAC '12). ACM, New York, NY, USA, 159-168. DOI=http://dx.doi.org.prox.lib.ncsu.edu/10.1145/2420950.2420976

[7] Alexander De Luca, Alina Hang, Emanuel von Zezschwitz, and Heinrich Hussmann. 2015. I Feel Like I'm Taking Selfies All Day!: Towards Understanding Biometric Authentication on Smartphones. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). ACM, New York, NY, USA, 1411-1414. DOI: http://dx.doi.org.prox.lib.ncsu.edu/10.1145/2702123.2702141

[8] Bishop, Matt, and Daniel V. Klein. "Improving system security via proactive password checking." Computers Security 14.3 (1995): 233-249.

[9] Abraham Kuri Vargas. "APIS ON RAILS Building REST APIs with Rails."

chits:
ddljuiau
fkjsuaie
fhhmueeo
bjjpeaia
dghyuiuu
djjsuaao
dkhgiiaa
dhjtuiua
dfjtoaao
dckbaoii
fgmmuual
gdbveoai
flgwaoiu
cdclioai
dlmvueio
dmkvuaea
fjjlaiee
fdgfeiua
fbjwueou
ffgveeei
fmksueii
bljdiioa
fclqaiei
cfhfoeei