ON COMMUNITY-BASED

AUTHENTICATION FACTOR

By

KHALID HAMMAD ALKHATTABI

B.S. C.S., Taibah University, 2008

A thesis submitted to the graduate faculty of the

University of Colorado at Colorado Spring

In partial fulfillment of the

Requirement for the degree of

Master of Science in Computer Science

Department of Computer Science

Fall 2015
This thesis for Master degree by

KHALID H. ALKHATTABI

Has been approved for

Department of Computer Science

By

______________________________________________
Dr. C. Edward Chow, Advisor

______________________________________________
Dr. Jia Rao, Committee Member

______________________________________________
Dr. Jonathan Ventura, Committee Member

Date…………………………………
Abstract

In this thesis, we explore the use of community Information as a factor for authentication and authorization, called it Community-based Factor (CBF), and present the design and implementation of a Community-based Authentication and Authorization (CAA) System which utilizes CBF. CAA is a system that enables a user to generate a CBF from the people of his/her community group for authentication and authorization purposes. CBF has been used in inherence granting scenario, court hearing, and referral in our daily life. One goal of the research project is to investigate whether the critical CBF information can be gathered in real time over today’s mobile networks and internet. CAA is a system, which on receiving an access request, can send SMS, utilizes Apple Push Notification service (APNs), and email to a group of pre-designated users for access approval. The weights of the approval can be adjusted based on the location proximity of the users, APNs-OPT, and Facebook SSO. Experiments are conducted among a group of colleagues to evaluate these different methods for their effectiveness. The results show that APNs is most secure and efficient, followed by SMS. Email does not meet the requirement. Our results found that the minimum time to login in real time with CBF is around 72 seconds when requiring authentication from two group members. The response rate is very important for CBF. With 16 participants in our experiment, the best result we got is 60% of participants respond within one minute or less. These preliminary results show CBF data can be gathered in real time which contradicts to our original hypothesis. We believe the usability of CBF for authentication in real time can be improved with the growth of smart phones and the behavior of the mobile users. In this
project we have developed both the CAA server and iOS mobile app for CAA clients. These software programs will be made available to facilitate the future research in CBF area,
To my parent, my wife, my Son
Acknowledgements

I want to thank all the people who helped, encouraged, and supported me to make my dream become a reality. First of all, I would like to thank my family overseas, my dad, mam, brothers, and my sisters for their support and for their love. Also, I would like to extend my thanks to my family here at Colorado Springs, my wife (Nadia) and my son (Kenan) for being patience, understanding, and acceptance of my being a busy student spending most of his time doing research. I especially would like to thank them for their love, support, and encouragement that they consistently provided to me.

I would like to express my gratitude to my advisor, Prof. Edward Chow for all his advice, assistance, patience, and encouragement to me during my efforts on this thesis. I learned much from him during my tenure in his classes as well as during my research. I am also deeply grateful to committee members, Dr. Jonathan Ventura, and Dr. Jia Rao for their comments, suggestions, advice, and guidance to succeed in this effort. I would like to extend my gratitude to Dr. Dr. Chuan Yue to serve on the committee for my proposal before he left the UCCS. I would like to express my appreciations to all my professors who imparted so much wisdom and help to me at the UCCS. I am really blessed to have learned so much from them during my time at the UCCS.

I also cannot forget to express my great appreciation to the University of Tabuk which provided an opportunity to me to extend my study overseas through their funding support. Finally, I would like to thank my advisor at Saudi Arabian Cultural Mission (SACM) for his unwavering support during my study tenure in the United States.
TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION ........................................................................................................ 1

1.1 Traditional authentication ..................................................................................... 1

1.2 Two-Factor Authentication .................................................................................... 2

1.2.1 Wireless attack: .................................................................................................................. 3

1.2.2 Mobile Phone Trojan: ....................................................................................................... 4

1.2.3 SIM Swap Attack: .............................................................................................................. 4

1.3 Three-Factors Authentication .......................................................................... 5

1.3.1 Fourth-Factor Authentication ..................................................................................... 6

1.3.2 Multi-Factor Authentications ....................................................................................... 6

1.4 Kind of Factors on Authentication .................................................................... 7

1.4.1 Knowledge Based Authentication ............................................................................... 8

1.4.2 Object Based Authentication ......................................................................................... 8

1.4.3 Major Identity – ID Based Authentication ............................................................... 9

1.5 Research Problem ................................................................................................... 11

1.6 The purpose of research ........................................................................................ 12

1.7 Need for Community-Based Authentication .................................................. 13

1.8 Challenges of Proposed Research ...................................................................... 13

1.9 Literature Review .................................................................................................... 14

CHAPTER 2
OPT OVER APPLE PUSH NOTIFICATION SERVICE ........................................ 18

2.1 What Is Apple Push Notification Service (APNs)? .............................. 19

2.2 Using OPT Over APNs: ........................................................................ 28

CHAPTER 3

THE DESIGN OF COMMUNITY-BASED AUTHENTICATION AND
AUTHORIZATION (CAA) ................................................................................. 31

3.1 The Proposed CAA .................................................................................. 31

3.2 Operation of CAA ..................................................................................... 33

3.3 Components of CAA ............................................................................... 34

3.3.1 Mobile Main interface CAAMI ............................................................. 35

3.3.2 Mobile group member interface GMI ............................................... 36

3.3.3 Authenticator 1 .................................................................................... 36

3.3.4 Authenticator 2 .................................................................................... 37

3.3.5 CAA component .................................................................................. 37

CHAPTER 4

IMPLEMENTATION AND TESTING CAA SYSTEM .................................. 41

CHAPTER 5

TRUSTWORTHINESS ON MULTIFACTOR AUTHENTICATION WITH CAA
SYSTEM ........................................................................................................ 47

5.1 Limitation of Factors in Factors box in Our Model ......................... 56

5.1.1 Location accuracy ............................................................................... 56

5.1.2 Limitation of Sending SMS by Carrier ........................................... 57

CHAPTER 6
CHAPTER 7

LESSONS LEARNED

7.1 Research Experiences

7.2 Build a New Skills to Future (Swift Language)

7.3 Discover many knowledge about multifactor authentications

7.4 Challenge of Mobile Computing

CHAPTER 8

FUTURE WORK

8.1 Improve the CAA Server Components

8.2 Framework to test mobile app based on app category

8.3 Machine learning and data mining to ask question from social network

8.4 Virtualization to Simulate Mobile Devices

CHAPTER 9

CONCLUSION

REFERENCES
APPENDIX A. ................................................................................................................ 81

APPENDIX B. ............................................................................................................. 100
Figures Lists

Figure 2-1 Overview of all the Tools that are necessary to configure APNs Technical. 20

Figure 2-2 One-Way Secure Communication between Provider and IOS device ....... 25

Figure 2-3 An initial connection between IOs device and APNs[16] ......................... 26

Figure 2-4 Interactions among APNs, Device, and Provider to share Device Token[16]
........................................................................................................................................ 27

Figure 2-5 Generates the Device Token and shares it with provider[16] .................... 27

Figure 2-6 Provider sends notification to IOS device by passing payload information
with a device Token to APNs[[16] ................................................................................ 28

Figure 2-7 APNs-OPT Overview Scheme ................................................................. 29

Figure 3-1 A sequence diagram for CAA System. Community-based factor performs
through the cooperation among three parties User, CAA server, and community group
members ..................................................................................................................... 32

Figure 3-2 Overview of CAA system. APNs server is used to distribute the OPT
through a secure channel ......................................................................................... 32
Figure 3-3 Main Components of CAA system, where Main App is CAA mobile interface, and CGM app is Community Group Member app.

Figure 3-4 CAAM interface app and Community GM Interface app.

Figure 3-5 CAA component, Sensor Module, and Making Decision Modular MDM, CAA database sever, and APNs server.

Figure 4-1 Mobil App for CAA system V.0. The left picture is a main interface, the middle is registration interface, and the right picture is SSO with Facebook to important information from Facebook.

Figure 4-2 A sequence diagram among CAA Server, APNs, and Community group members.

Figure 4-3 Mobil App for CAA system V.0. The left picture is a main interface, the middle is registration interface, and the right picture is SSO with Facebook to important information from Facebook.

Figure 5-1 Our visual of trustworthiness policy rules on MFA based on CAA.

Figure 5-2 Community Group Member CGM app used for voting by a group member.

Figure 5-3 Many of social network websites offer Single Sign On (SSO) services.
Figure 6-1 Comparison among different methods to deliver the OPT. APNs-OPT transfers the OPT over APNs server by using remote notification is much faster than the others.

Figure 6-2 Comparison of Responding Curves for SMS, APNs, and Secure EMail with Community Group Member of 10.

Figure 6-3 Comparison among Traditional authentication, OPT over APNs, and Our Community base- Factor.

Figure 6-4 Storage needed by our CAA mobile app on IPhone device.

Figure 6-5 Behavior Current location of Community Group Member.

Figure A. 1 Create A new Project

Figure A. 2 Application Template

Figure A. 3 Product Name and Bundle identifier for IOS APP

Figure A. 4 Apple ID and App Secret for Facebook App

Figure A. 5 Register your app ID in facebook developer

Figure A. 6 Info.plist path
Figure A. 7 Info.plist’s attributes ........................................................................................................ 92

Figure A. 8 Three attributes are needed to create in info.plist....................................................... 93

Figure A. 9 To add bundle Identifier for your App................................................................. 93

Figure A. 10 Bridge-Header between Object-C and Swift Languages. ................................. 94

Figure A. 11 Bridge-Header between Object-C and Swift Languages, Facebook SDK. ................................................................. 95

Figure A. 12 To tell Swift Compiler about Path of Bridge-Header.h. ................................. 96

Figure B. 1 Explore Xcode's User Interface.................................................................................... 89

Figure B. 2 Adding App ID Account in XCode Preferences.................................................. 101

Figure B. 3 Adding Apple ID Account in XCode Preferences.................................................. 102

Figure B. 4 Existing certificates and provisioning profiles..................................................... 102

Figure B. 5 mfauth Icon when it deployed on Iphone device ............................................ 103

Figure B. 6 CFA interface ........................................................................................................ 104
Figure B. 7 Registration Interface need User's information, and a group members information ................................................................. 105

Figure B. 8 Add More Friends Interface .................................................................................. 106

Figure B. 9 Import From Facebook Interface ........................................................................ 106

Figure B. 10 mfauth Icon when it deployed on Iphone device. Error! Bookmark not defined.

Figure B. 11 SMS contents which will send to all a group members. Error! Bookmark not defined.

Figure B. 12 Voting page contents. The picture is imported from Facebook .................. 107
Chapter 1 Introduction

Authentication in cybersecurity is a mechanism employed to compare a current users credentials with what was previously stored to either allow or reject user access to local or remote resources. The users credentials is required information through which the credential software decides whether or not a user is legitimate. Authentication is one of the important parts of cyber security in term of identifications. Authentications, and cybersecurity, play an important role in our life today due to the growth of remote services. Unfortunately, authentication is the weakest component of cybersecurity. It is the primary target of hackers. One of the most popular authentication forms is traditional authentication, the use of username and password for validation.

1.1 Traditional authentication

Traditional authentication is the most common and primary way to authenticate individuals by requesting and validating a username and password. A username is a unique identifier for a user, and the password is a hopefully secret piece of information
created by the user, and, hopefully, one that is easy to remember. Traditional authentication has several characteristics: simple of implementation, low cost, and low executing time. A simple authentication sequence begins when a user is required to provide his/her credential information to obtain access to a secure resource on a secure server. A remote server should verify username and password with the information it has previously stored.

Today, traditional authentication methods that verify authentication through username and password have become inadequate for most authentication purposes. Username and password can be forgotten, disclosed, lost, or stolen. A username is more easily obtained by hackers or other unscrupulous users than passwords. However, passwords can be predicted based on the user’s profile, or through sharing the same devices or workstations by coworkers. Through analyzing patterns, a GPU or brute-force algorithm, they can quickly be discovered. Careless user habits also lead to passwords being transcribed to and easily discovered on notes, files, or similar resources. Traditional authentication is plagued by several types of cyber-attack such as reply, eavesdropping, and phishing, any of which can easily be prevented through the utilization of Two-Factor authentication.

For example, On Oct 14, 2014, Dropbox was compromised by hackers who posted usernames and passwords on pastebin.com for more than 400 users[1]. To remedy their vulnerabilities, Dropbox enabled Two-Factor Authentication to protect user accounts. [1] .

1.2 Two-Factor Authentication
Two-Factor Authentication (2FA) adds an extra security layer of authentication compared to traditional authentication. 2FA remedies the vulnerability that comes from using the traditional authentication alone, such as reply attack and phishing attack. Any combination of two factors forms a 2FA system. Usually, 2FA uses a One Time Password (OPT) as the second factor.

Many Studies have proposed authentication by utilizing OPT based on Short Message Systems (SMS), Biometric, QR-Code, or public key, etc. [2][3][23][4]. Today, many technological methods of communication are utilized to transmit, OPT such as SMS and E-mail[3]. One of the most common OPT delivery methods is SMS[5]. The simple concept flow behind SMS-OPT begins when a user enters his/her username and password to log into a secure system. The server generates OTP which then sends the information to the user through an SMS message. The user receives OPT on his/her mobile device to use it for log in. One core issue with using SMS-OPT is a transmission delay related to SMS and roaming service[4]. Additionally, the cost of using SMS is high compared to E-mail. Moreover, many studies have demonstrated that SMS-OPT is not secure. There are several possibilities of attacks for breaking confidentiality of using SMS-OPT. In the next sections, we briefly review common attacks [5][23].

1.2.1 Wireless attack:

There are several types of attack available on wireless networks, such as A/1, A/2, due to the use of the network’s weak encryption algorithm. Although there are variations of the A/x encryption family which are more secure than A/1,A/2, such as A/5 and A/8,
A/1 and A/2 are still commonly employed according the article written by Craig T. and Ashkan S., “NSA able to Crack encrypted calls, texts”. Attackers exploit the weakness of an encryption algorithm to intercept all network traffic between stations. In cellular network, technology such Global Service for Mobile communications (GSM) is used in world-wide mobile services to provide variations of services such as Calls, VoIP, and SMS. Many studies proved that GSM is not secure due to the encryption algorithm utilized [6] [7] [5]

1.2.2 Mobile Phone Trojan:

Malicious mobile apps are unintentionally deployed by a user on his/her mobile device while downloading games or other apps. Malicious apps can contain worms or malware that harms the user’s device. Some malware apps are able to intercept SMS messages. In [5], a list is presented some Trojan apps which intercept SMS-OPT for hackers. These include ZITMO (Zeus IN The MObil) and ZeuS app for Windows Mobile.

1.2.3 SIM Swap Attack:

The SIM Swap Attack is a social engineering attack that happens when an attacker attempts to request a new SIM card from mobile service operator for a user (victim) and transferring current user’s mobile number to another one[8]. This deactivates the victim’s SIM card and activates new SIM card belong to the attacker. Then the attacker can intercept all calls and SMS sent to the victim [5][23].
In this research, we will use an alternative method to send OPT over Wi-Fi connection with Apple Push Notification service (APNs). We will explain in more details in Chapter 2.

1.3 Three-Factors Authentication

Even with two-factor authentication, the authentication system can still be broken through eavesdropping attack, man-in-middle attack, guessing OPT, or stealing a token-generating device. For these reasons, researchers propose the enhancement of the two-factor authentication with a new scheme called three-factor authentication.

Three-factor authentication (3FA) provides one extra security layer over 2FA. A 3FA usually uses a biometric factor as a third factor. Biometrics are features that can be used to distinguish humans from machines, such as iris, face, retina, fingerprint, etc. Using biometrics for identification is more efficient than password, OPT, or the combination of two factors. Biometrics is often used by governments to identify individuals because of its efficiency. Many studies proposed 3FA by integrating biometric factor with 2FA schemes. Hyun-Sung K. [9], proposed a 3FA as ID-based Password authentication system with smart card and fingerprint.

Definition: The use of three factors to identify, i.e., authenticate, a user. The three factors are: something the user knows (e.g., password), something the user has (e.g., token), and something physical from the user (e.g., biometric).

1.3.1 Fourth-Factor Authentication

Social Authentication has been involved in the authentication process. Brainard et al. proposed fourth factor, e.g., somebody known, to authenticate users[10]. The idea behind his proposal is that a user who loses his primary password or token asks a pre-select ‘helper’ to assist him to access a system or recover a password. His proposal was combined with two-factor authentication (PIN and SecurID Token) [10].

1.3.2 Multi-Factor Authentications

Multi-factor authentications (MFA) have become a new trend in identifying authorized users and to remedy many vulnerabilities in cyber-security through adding extra layers of authentication. In the last decade, the growth of remote services over the Internet increased. Simultaneously, the threats also increased. Therefore, more layers of authentication such as MFA are necessary. MFA requires systems to provide more credential factors for the authentication of individuals. The types of factors available on MFA environment include password, One Time Password OPT, fingerprint, face, voice, user location, Graphic Image, International Mobile Subscriber Identity (“IMSI”), etc. [3]. There are pros and cons for each factor in terms of performance, computations, speed, memory, cost and weight. In the next section, we will explain Multi-factor authentication in more detail.

One of the common instances of using MFA in our community is inheritance granting. To provide authorized rights to a person to obtain their estate, he must identify himself by providing probate document to organizations, banks, work, etc. In some
cultures, to have a probate document, a person needs to identify himself by presenting several pieces of identification. If one piece of identification is missing, the person will not be authorized to have a probate document to allow him legally possession of his/her estate. For example, when there are siblings who attempt to receive a probate document which confirms a legal right to their estate, they must attend and provide their identifications and stamps. Moreover, witnesses might sometimes attend as well to identify the siblings to inheritance court or the organization that issues probate documents. The witnesses are a legal trust who know a both the deceased person and their heirs, such as friends of the deceased. We found that a community can participate in the authentication of individuals in real life. This is our topic of our thesis community-based factor.

1.4 Kind of Factors on Authentication

Multi-factor authentication MFA is an environment in which numerous factors are available for user authentication. Today, there are many factors in multi-factor authentications. O’Gorman summarized them in three main types[11]:

1. **Knowledge based** – “Something you know?” It depends upon user’s own knowledge of keeping the password

2. **Object Based** – “Something you have?” These are actually related to physical objects that are used for the authentication of a specific user and must be with that user attempting authentication
3. **Major Identity** – “*Something you are?*”. This is basically a user’s own major ability of identifying himself/herself to the computer with one unique personal thing such like biometric identification.

Certain other things, like the main authentication techniques, are also considered as authentication methods, but these all fall under the above mentioned factors. O’Gorman detailed the classification of Authentication factors as follows:

**1.4.1 Knowledge Based Authentication**

Knowledge based factor stands for the information that is the user’s responsibility, whether it is a super-secret information (*Something you know*) that is within the user’s knowledge or information that is intended to be provided to the system for authentication at that time. Usually this involves passwords or PIN’s that are kept secret and are memorized by the user and are to be used by the user only. Username and password is classified as knowledge based authentication factor or one-way authentications factor[11].

**1.4.2 Object Based Authentication**

Object based authentication requires a basic object (*something you have*) that is within the possession of the physically present user to allow authentication. These are actually tokens of identification, and in most of cases, these objects provide physical evidence of the user’s authenticity to the system. This includes ATM Card, ID card, a metallic key, a boarding pass, etc. One such example is the computer’s USB stick that enrolls a digital signature for authentication within the Operating System allowing the user
to log in using physical token identification[11]. Such identification inside the system helps maintain the security. Even if one safe code is lost, the user can claim the security by providing the physical presence of the secure object. This can include a one-time-password, as provided by most of the online servers, to send a randomly generated authentication code. After authentication, that code is flushed. It can also be some other form of authentication[11][4].

One such example of the physical existence of a token is AES authentication for certain tasks using a vault[11]. This method AES (Advanced Encryption System) involves saving a password of fixed 256b into a vault. As the human memory is incapable of storing such large passwords, the password as a PIN, provided to the vault by the user, is thus entered to the vault with encryption phenomena that directly converts the password to the standard length of AES. For this instance, if there is a hacker who wants to guess the value of that password, the system will require $10^{76}$ guess values for the password at the best computer speed available today. This is by far the least physical existence of the password, but the password that is provided to the vault by the user is also a guessable password. Once an attacker’s access to the vault becomes feasible for a certain user, the attacker can guess and break the user’s password. But such password on a vault cannot be decrypted.[11]

### 1.4.3 Major Identity – ID Based Authentication

This type of authentication comprises the unique factors of a given user. This mainly consists of an authentication of a body part or other major unique features. This
can be license authentication, ID card authentication or it can be purely biometric. Biometric authentication involves the authentication of various parts of a human being that the system employs directly to authenticate the user. In this method, fingerprints, eye scan, voiceprint or signature can be used for authentication due to the unique features of each[11].

A fingerprint has a unique value for every user, as does the eye scan. Voiceprint is actually a best option as pitch and certain other factors related to voice of a certain human being cannot be matched to any other person. Signature checking is a graphical authentication which involves the verification of a user by entering a signature or other personal graphics that can be employed and that remain unique to that specific user. One drawback related to such authentication is that, if the user has lost that key to identification, e.g. eye surgery or hand cut, etc., this authentication process can never be rolled back either by normal methods or through the device’s coding. Therefore, this manner of authentication is often rolled together with another authentication technique to ensure no loss of identification validation for the user.[11][12]

In this thesis, we explore the design and evaluation of a new authentication factor that relies on community groups to authenticate individuals within the community, based on their association or trust in a multi-factor authentication environment. The CAAS server is utilized for registering users who are willing to authenticate others through their smart phones in real-time through the texting and OPT or E-mail and OPT. An authentication client module can then query the community based server to see if a
member is trustworthy based on voting and current location of friends and users, which then uses them for authentication purpose.

I compared this new factor with the existing multi-factor authentication such as geolocation and OAuth-based authentication from social network sites.

1.5 Research Problem

We investigated the following research questions:

Q1. How can we weight the trustworthiness of multi-factor authentication?

A multi-factor authentication system is required to collect, save, and verify multiple authentication factors for each individual user. Some factors can be collected with the active interaction of users. Others can be collected and verified without the user’s involvement. Certain weight of trustworthiness can be assigned to these factors based on the credibility of the information provided - the time, place, and the manner in which we collect them. The existence of one factor may help boost the trustworthiness of another factor. The Geolocation or the OPT can significantly raise the trustworthiness score of combined factors. How to assign these weights and adjust them based on context and availability is a challenging issue. Depending on the environments on which they are operated, multi-factor authentication systems could assign different weights to factors and use only them whether other factor values exist or change. In extreme cases, the weights will all be zeros until all factors are available. How to apply policy rules for computing the overall trustworthiness or implementing a program to take context/history information into
consideration is a design trade-off question. One of the metrics for evaluating the effectiveness of MFA is the reliability of trustworthiness calculation. The requirement for a MFA using casual web server access vs. that of an inheritance granting system.

**Q2.** Can a community-based authentication be effective as a key technology for multi-factor authentication?

We will discuss all related topics to this question such as the privacy and security of using this factor later in this thesis.

**Q3.** Can we consider a location of my friends or group member’s locations as a factor?

According to [13], the author agrees that current location can work as a passive factor in the authentication process. Active factors or user interaction can be reduced by using passive factors such as current location or time of last log in.

### 1.6 The purpose of research

The purpose of the study is to create a new factor for community-based factor or using group people in authentication as one of the factor in MFA, either asynchronous or synchronous, to improve current authentication systems with mobile devices, and to investigate whether these new factors can effectively enhance cyber security systems.
1.7 Need for Community-Based Authentication

In some cultures, people employ multi-factors to identify an individual based on community. For example, in Inheritance court in Saudi Arabia, a person is identified by a group of people in addition to his social security number. Of course, the group of people must identify themselves as well. In this example, we can see how the community can help to identify the individual. Early in the wars, the military was used as a group of people to identify the military members as a means of detecting intruders. With the proliferation of mobile phones, social media can afford rapid communications with group members. Therefore, we can use this principle over applications such as online courses, interviews, and in courts.

Since we live in groups, there is a least one group who can help to identify us. In fact, my wife and I are a group; my brothers and I are a group; my friends and I are a group; my classmate and I are group, etc. There are many groups in our community which can be created. We need to employ this group advantage to enhance authentication in applications. We need to use this benefit over multi-factor authentication by proposing a new community based factor for MFA. We need community members participate to identify and detect intruders over applications.

1.8 Challenges of Proposed Research

How to weight and choose a combination of factors for effective authentication?
The use of community relationship in authentication is new and very little software implementation is available. How can we apply it over current applications in real time?

1.9 Literature Review

A group authentication is a new authentication type proposed for group-oriented applications. A group communication takes the place of multiple communications where many users to communicate to each other, often simultaneously. In a group-oriented application, each user has to authenticate other users to allow them to communicate within the group.

There are two methods of authenticating users in this application, either centralized or non-centralized authentication. In Lein Harm, 2013, he proposed the manager is responsible for registering all group members. After all the members are registered, the manager will generate a token for each user. The group members use this token to authenticate themselves. [14][15].

Moreover, in October, 2011, Facebook enabled Social Authentications for when the user loses his password by showing pictures of some of his friends and letting the user identify them.

A fourth factor authentication 4FA is proposed by[10], it is a Category of “somebody you know” factor. As proposed, 4FA is working as an emergency Authenticator. An emergency Authenticator is Authenticator used in emergency cases, such as forgotten a password, or a token which is a primary factor that must be provided to
the system to as part of a login. The most common emergency Authenticator is E-mail sending a temporary password to user’s E-mail registered in user’s account. The scheme that is proposed for 4FA depends on a Vouching term. Vouching is a peer-level authentication as asker and helper. The vouching system is used when a user (the asker) request help from helper to obtain a temporary vouch code in terms of assisting him to log into the system. The proposed system uses 2FA based on PIN, which is a password and SecureID hardware device which generates a token every sixty seconds.

Our proposed system is different from what has been proposed:

- Our system replaces a helper to Community group members or trust group members

- Our system utilizes trust group members from a community which is around the user and is a primary Authenticator in real-time to log into a system which is different from what was proposed as emergency Authenticator.

- Usability of utilizing helper as emergency Authenticator is improved by using a mobile phone to help a user to log in. People are always in possession of their mobile phone.

- It is not necessary to maintain communications between the user and helpers to exchange a temporary password. Our system will communicate directly to user and Community group members. A CFS server will verify the helper by sending OPT as verification letter by using an APNs
- Our system community based system is integrated with six another factors to be a low-weight risk assessment such as Friend’s location.

- Our system is using a group community location as a system factor for either verifying a community group member or to comparing to the current user location.

Our proposed system is classified under social authentication [10]. We propose a community-based factor CBF. A CBF based on a voting and current location of Friends. A voting is used to identifier a user to the system by one person or a group of trust members in real time. For a current location of friends, since a current location of user is considered an effective factor in multi-factor authentication [13], it will be a trustworthy factor for our system for community group members as well.

In our system, we will use the current location of friends as a factor by determining if the current location of the user matches with the current location of community group members. If the location is matched, that means the user has a higher possibility to be a legitimate user. Our community-based factor system depends on a pre-selection of a group of friends who will help user to identify himself to a system by voting.

Moreover, System will retrieve some information from Facebook about group members (close friends) such as have we met together somewhere? Have we tags together? The answer for these question will be in initial stages. In addition, I will receive information about the group’s member locations from where they are voting. A current
location of a user is considered as an effective factor because people spend most of their
time either at home or work [13]. Our question is what about the remainder of the time?
Where do they spend it? Of course, they hang out with their close friends either in
restaurants or in places that are frequently visited. That is the purpose of taking the current
location of friends. In addition, we would like to know who is around a user in real time. If
the people who are around the user are his/her group member, the possibility of the user to
be legitimate is high.

Moreover, as we mentioned in Section 1.2 that SMS-OPT suffers from many
security and technical issues, we are going to use a notification message to transmit the
OPT to the mobile device user instead of SMS-OPT.

In this thesis, we investigate the design and implementation of a community-based
authentication system (CAAS). Chapter 2 discusses the use of OPT Over APNs. Chapter 3
presents the design of a CAAS. Chapter 4 describes the implementation of CAAS. Chapter
5 discusses the evaluation results of CAAS. Chapter 6 shares the lesson learned in this
project. Chapter 7 shares my thoughts of the future directions of the research area. Chapter
8 is the conclusion.
Chapter 2  OPT Over Apple Push Notification Service

One Time Passwords (OPT) is a password system that is generated by software or hardware for one-time use. A one-time password can be generated by a variety of ways, either as software (as an algorithm) or hardware (as a physical device). Many schemes have been proposed to generate OPT by using hashing functionality, Smart Card, using GPS location, seeds, IMEI and IMSI numbers as the seed to generate the OPT [3][4]. We discussed OPT in more detail in Section 1.2

Today, there are many forms of communication technologies that help people communicate together by either asynchronous or synchronous ways. For example, E-mail, Social network, SMS, Online Chat, social Apps, etc., are forms of communication technologies. Recently, the primary goals of these communication forms have changed. They became used for many purposes, such as marketing and security. In marketing fields, many businesses owners use them to sell their products by sending messages to a large number of customers. In the security field, some of those communication technologies are
used to authenticate people, such as SMS, E-mail, and Signe one On (SSO). Many systems enable two-factor authentication by using SMS-OPT and E-mail-OPT as the second factor.

OPT is one popular factor for both SMS and E-mail. However, using SMS-OPT causes security and technical issues. In Section 1.1.2, we discussed many potential attacks can intercept SMS-OPT, such as wireless attack, and mobile Trojan. In addition, there is a delay limitation caused by using SMS which is out of the control of authentication system administrator. In 2009, Apple founded a new one-way communication, remote notification. The goal of remote notification to inform a client about an event which occurs on a remote server. To be more sophisticated, it is able to use a remote server to send notifications/messages to particular IOS devices. Apple Push Notification Service (APNs) is a central component of the Remote notification feature for IOS and OS X devices[16]. We will further define how APNs works later in this document.

2.1 What Is Apple Push Notification Service (APNs)?

APNs is a new kind communication technology that is popular on Apple devices such as IPhone and OS X. APNs provides a remote notification service that enables a provider (remote server) to send alerts or short messages to user’s device.

Figure 2-1 shows an overview of what a mobile developer needs to enable it to send remote notifications to a device.
A mobile IOS developer needs to register on developer.apple.com website to create an account. As soon as the account is created, he is required to do two things: request a Signing Identity and create App id for an App. The Signing Identity consists of a public and private key as a pair which is issued by Apple[16]. The public and private key needs to be stored on keychains on the laptop/PC which hosts the Xcode project. There is another certificate called intermediate certificate which needs to request the Signing Identity. This certificate will be stored to keychains on the Xcode hosting laptop/PC. The role of Signing Identity (Public and Private key pair) are used to sign your app into the building process[16]. In the building process of the app, the operating system can verify your app based upon a signature on your app to determine whether your app is modified or
not. The signature is made by a certificate which had a just public key and is stored on the developer account[16].

For APN service[16], every app needs to use a Push Notification service; it has to have three things:

1. It has to have a unique *App Id* which is the same bundle identifier on your Xcode project for the enabling of Push Notification service.

2. Create APNs SSL certificate for the unique *App Id*.

3. Registering your app to remote notification by using the following code at `AppDelegate.swift` file:

   PHP script to sending A notification to APNs

In order of enable remote push notification to CAA app and CG Client App, we follow the next steps[17]:

1. Generating Certificate Signing Request CSR from certificate authority which is Apple. After the Certificate is generated, private key `PrivateKey.P12` will be available to download and export. You need to store it at safe place for future.

2. Generating two SSL Apple Push Notification services certificates Cer1.cer and Cer2.cer, one certificate for each app.
3. Save what you generated CSR, PrivateKey.P12, and Cer1.cer in director1 and CSR, Private key PrivateKey.P12, and Cer2.cre in director2. Director1 is for CAA App and Director2 is for CG Client App.

4. In order of using PHP scripts to sending a payload to particular app which locate on device, we need to convert the SSL certificate for APNs that we generate Cer1 and Cer2 to PEM files.

   $ openssl x509 –in Cer1.cer -inform der -out Cer1.pem

5. Also, we need to convert the Private Key to PEM file:

   $ openssl pkcs12 -nocerts -out privateKey.pem -in PrivateKey.P12

6. Then, we need to combine both Cer1.pem and PrivateKey.p12 to generate single PEM file which the server will use it to connect with APNS every time need to send message.

7. PHP script that we used for connection with APNS[17]:

   // Device Token is device address, which our app is locate on it
   $DeviceToken = "Hexadecimal numbers ";

   // Our function to sending Notification

   Function SendNotification($Message, $deviceToken, $Certificate,$ExtraValue){

      //Passphrase for combine PEM file

      $passphrase = 'XXXXXXXX';
// Message that we attempt to send it
$message = "$Message";

// Set Context options include path of certificate and passphrase;

$contextOptions = [

    'ssl' => [

        // certificate path

        'local_cert' => $Certificate,

        'passphrase' => $passphrase,

    ],

];

// Creating Stream Context with set context options

$StreamContext = stream_context_create($contextOptions);

// To open a Socket with APNs

$fp = stream_socket_client(

    'ssl://gateway.sandbox.push.apple.com:2195', $err,

    $errstr, 60, STREAM_CLIENT_CONNECT, $StreamContext);

$payload['aps'] = array(

    'alert' => $message,

    'sound' => 'default'

);

$payload['message'] = "$ExtraValue";
```php
// Simple APNs format notification used to

// Pack function to represent the data in the binary structure

$payloadJSON = json_encode($payload);

$SendingMessage = chr(0)  // 1 Byte with 0 value
    . pack('n', 32)         // 2 Bytes endián byte order for Device token length
    . pack('H*', $deviceToken)  // 32 bytes for token
    . pack('n', strlen($payloadJSON))  // length of payload
    . $payloadJSON;

// Sending payload to APNs

$Response = fwrite($fp, $SendingMessage, strlen($SendingMessage));
```

The role of APNs is to provide two secure channels among three parts, a provider, APNs, and an IOS device. A provider is an end-point Server (remote server), APNs server, considered as a gateway for remote notification, and a device - a host an App. The communication between the three parties is using TSL (Transports Security layer) in one-way communication[16]. See Figure 2-2.
In the initial step [16], an app will generate a request to connect with APNs, APNs server will respond with its certificate to the IOS device to identify itself. The IOS device will validate a Server Certificate and respond with its Certificate to be validated by the APNs server as well. After both sides are verified, APNs will generate a unique Device Token identifier for IOS device. See Figure [16]
Device Token is device address that can be used to send remote Notifications too. Device Token is an encrypted Token using Device certificate. Every time the device connects to APNs, it has to provide its Device Token for verification by APNs server. It must be shared with provider in terms of sending a notification to that device the Device Token indicates. On the other hand, provider can establish the SSL connection with APNs by providing its certificate to APNs server to verify. If it is verified, APNs will respond with its certificate to verify as well[16]. When both sides are validated, the SSL connection will be established. See Figure 2-5 [16] show protocol interactive among device, APNs, and provider of generating Device Token in order to share it with the provider.
Figure 2-4 Interactions among APNs, Device, and Provider to share Device Token[16]

Figure 2-5 Generates the Device Token and shares it with provider[16]

Figure 2-5 shows that every time IOS device tries to connect with APNs, it has to provide its Device Token. The APNs will decrypt the token by using the Token key based on the Device certificate and will return validation[16].
When the server needs to send a notification to specific device, it must provide a Token with payload information to APNs. APNs will decrypt the Token and verify it and then send the notification to the device that holds that Token. See Figure 2-7 [16] for the details protocol.

![Figure 2-6 Provider sends notification to IOS device by passing payload information with a device Token to APNs][16]

### 2.2 Using OPT Over APNs:

Many studies demonstrated that the use of SMS-OPT experiences security and technical issues as mentioned in Section 1.1.2. A SIM swap attack can exploit a vulnerable mobile phone number by transferring the mobile number of a victim to another SIM card which an attacker has. The attacker will then capture all the calls and SMSs of a victim. Moreover, technical issues, such as transmission delay, cost [5][4] [SMS-Based One Time Password Vulnerabilities and Safeguarding OTP Over Network] can appear. SMS works based on a phone number which is available for everyone, and will make it a vulnerable for a swap attacker[8][5]. To remedy these security and technical issues, we use an alternative method to transmit OPT by using a mobile phone which will not depend
on a phone number. We are going to utilize a remote notification to send OPT to a user’s mobile device with SSL communication with the assistance of APNs. By using remote notification as the delivery method, we remedy the vulnerability of using SMS-OPT which depends on SIM card.

Figure 2-7 APNs-OPT Overview Scheme

Figure 2.8 shows the overview of using APNs-OPT. The basic idea behind of using APNs-OPT is that a user has an app on his IPhone that uses remote services, in terms that using the remote services requires a registration. In registration, a user will use his/her mobile device to create an account. After all his/her information is entered, the user will submit his/her account information to the remote server including a Device Token of his/her IPhone. In Login, a user will provide his/her primary credential information which is required to log in. After the server verifies the credential information provided by a user, the server will send OPT through APNs to the user device. The user
will receive OPT on his/her device by remote notification. The user who has the mobile device which is linked with her/his account is the only person able to receive the OPT. The OPT will send it by using SSL channel to user’s mobile device.

The OPT sends to the user by using a different connection, which means if there are eavesdropping attacks or man-in-middle listeners to communication, it will not drop the OPT when the server send it to user. However, the attacker can drop or listen in with just one connection, which can happen unless there is an SSL connection for both connection from the user to the server and from the server to the user which is already using SSL.

We can use symmetric encryption based on OPT that is secretly sent via secure connection. Assume we are using a symmetric encryption for login, and then a shared key can be shared between the user and server by using remote notification. By using remote notification, we can dynamically change the shared Key every time a user tries to log in and send it to the user by using SSL through APNs. We believe that we can use a remote notification in many scenarios, and schemas to employ a secure authentication.

In our community-based system, we use APNs-OPT which is efficient and works well. The delay time for both SMS-OPT and APNs-OPT was noticeable to us. The delay Time for SMS-OPT was more than the Delay Time of APNs-OPT.
Chapter 3 The Design of
Community-based Authentication and
Authorization (CAA)

3.1 The Proposed CAA

We propose a Community-Based Factor (CAA) system which can be classified under the social authentication category. We present a novel mobile app for a Community based Authentication and Authorization (CAA) system which, on receiving an access request, sends SMS directly, utilizes an Apple Push Notification service (APNs), or uses both of them to a group of pre-designated users to approve the access. We will take inheritance granting as an illustration and a basic concept of our proposal. We will see that the usability can be improved with the growth of mobile phone use today.

In CAA, every user in our system will select group members who will help a user identify him/herself to a secure system through voting. Voting can be accomplished by
sending SMS, or E-mail, and APNs-OPT to group members. See Figure 3.1 which shows a sequence diagram for our overview of CAA:

![Sequence Diagram](image1)

**Figure 3-1** A sequence diagram for CAA System. Community-based factor performs through the cooperation among three parties User, CAA server, and community group members.

![Diagram](image2)

**Figure 3-2** Overview of CAA system. APNs server is used to distribute the OPT through a secure channel.
In our design, we will send two messages to each group member. One SMS message contains a voting link, and the other is a notification message to their IPhones. A SMS contains an indirect URL to on a CAS server where the voting page is hosted. The notification message has a verification code to ensure the confidentiality of voting. Every group member votes either by Yes or No, and concurrently sends his current location. The current locations can be used to enhance the credibility of the votes with verification code. Figure 3.2 shows overview of CAA system.

3.2 Operation of CAA.

A user must provide a username and password as a primary factor. Based on username and password, CAA will verify a user whether or not he/she has community group members. If he/she does, CAA server will send SMS and notification messages to all group members for voting. The SMS has indirect URL. All users’ group members need to click on the link and access the voting page on CAA server. The voting page displays a question and picture of the user who is tried to access. We believe that alternative method can be used such as voice call and video call instead of using a picture of the user to verify user recognition or not. Every group member can vote, sending his/her current location with voting. A voting is to confirm if user is recognized or not. The current location of friends is needed to make a decision in some cases. For example, a user who had his wife as a community member goes to a restaurant to have dinner. When the user tries to log into the CAA, then the CAA server receives his wife’s vote, the CAA server will find that user close to his wife. The CAA will increase the weight of this vote and the probability of the user to be legitimate. For security, all group members should receive a notification
message on their mobile device which contains APNs-OPT (verification Code) used for submitting the vote. The purpose of using a verification code is to provide the confidentiality and integrity of voting. The voting should come from the right person.

A current location of user has been proven as an effective passive factor in an MFA environment[13]. Most people spend most of their time at home and work, and our question is what is about the rest of time where they spend it? Of Couse, they hang out with their friends or family. Most of the time, there is at least one of their group members or a trusted person we hang out with. The system tries to discover who is around the user in real time. If at least one group member is around the user, such as user’s wife, then the current location of user should be matching with current location at least one trust group member. In some cases, it can be all group members are around the user. Therefore, possibility and probability of legitimate of user is increased. Moreover, a person with their friends always hangs out at somewhere that frequently visited.

3.3 Components of CAA

A CAA system has six main components as displayed in Figure 3-3. Each component interacts with others to preform one or more tasks. The main components are mobile Main interface CAAMI (App 1), mobile group member interface GMI (App 2), Authenticator 1, Authenticator 2, APNs server, and CAA server. See Figure 3-3 that shows overview picture of CAA system components.
3.3.1 Mobile Main interface CAAMI

CAAMI is the main interface for our app that enables users to make a request to the CAA system. A user uses this interface for either registration or authentication. The user will not launch any other interfaces on our app or access any resources on CAA the server unless he is the authorizing person. In order for a user to be an authorizing person, he/she has to provide credential information to the Authenticator 1 component. The authenticator 1 component response to authenticate a user. We will discuss Authenticator 1 in the following sections. See Figure 3-4.
3.3.2 Mobile group member interface GMI

GMI is a portable mobile app for voting purposes. The GMI app is deployed on all group members’ IPhone Device. The voting interface cannot be launched unless the Notification message (with permission code) is sent to GMI app. As soon as the user opens the notification message, the voting interface will be launched and it enables a group member for voting. In addition, in order to send a vote, a group member receives a SMS as well. The SMS includes a verification code that enables a group member to vote or send his/her vote. GMI is dealing with the Authenticator 2 component. Authenticator 2 is responsible to verify a voting. We will discuss more about Authenticator 2 in the next section. See Figure 3-4.

3.3.3 Authenticator 1
Authenticator 1 component is responsible for authentication users. Every request from a user must go through this component. Authenticator 1 has many tasks to perform. First, it is responsible for initial authentication of user when a user provides his/her credential information such as username/password. Second, it is responsible to keep track of each session’s authenticated users. Third, it is responsible to answer user requests. Fourth, it is responsible to retrieve or store data to the CAA Database. Fifth, it is responsible to give permission to launch any interfaces on CAAMI or any other resources on the CAA server. Finally, it is responsible for intermediate components between GAAMI and the CAA server.

### 3.3.4 Authenticator 2

Authenticator 2 component is responsible to compare a verification code of a group member that should be received by SMS with what is stored in the CAA server. If they match, Authenticator 2 accepts the voting information and sends a request to store it. In addition, Authenticator 2 acknowledges a group member if his/her voting is accepted or not accepted. Moreover, for each accepted voting, Authenticator 2 stores the voting information to the CAA database, and notifies a making decision module on the CAA server to remake a decision based on what information has been received at that point. Also, the making a decision module notifies Authenticator 1 about a new decision made to notify a user. We will discuss more about the task of making a decision in the next section.

### 3.3.5 CAA component
The CAA component has many modules, such as sensor module, making decision module, and community-based factor module. In addition, it is connected with the database server, and with APNs server.

A sensor module is responsible to receive or send a request from the internal module to Authenticator 1 or Authenticator 2. It is a panel control of CAA system. See Figure 3-5.

Making Decision Module MDM is the center of our design. It is responsible for making decisions based on what information the CAA server has about a user. Moreover, making decision will depend on a page ID and related permission code that the user requests. Each page has a permission code for access and which distinct kind of privileges are needed to access. The permission code is similar to what is in Linux. In Linux, each file or directory has a permission code which presents the privileges necessary for a user to access. Each permission code has policy rules that must be applied in order to give access rights to a user. CAA has many policy rules for each permission code. Moreover, policy rules can be applied based on the amount of information a making decision module has regarding the user who seeks access to the page. Example: If we return to our model illustration community base factor, which is inheritance granting, to give authorized right to a person to get theirs estate from banks or other organizations, a person must have a probate document to get rights access to estate. The person must provide the probate document to banks or other organizations, and based on probate document information, the banks apply his policy rules. To have probate document, a person has to authenticate himself by ID, social security number, or provide all relatives he has to present. MDM
will generate probate document for the user for each page ID he requests. MDM generates a base on what policy rules can be applied and how much information it has about the user at that point of decision making. See Figure 3-5.

Example: A user requests access to a page ID “bank account” that has a permission code XXX. A policy rule for XXX is “accept incoming request for bank account page if a group member of user is voting yes”. Authenticator 1 will receive the username/password of the user for the initial authentication of user. If a user has an account and the username and password is matched, Authenticator 1 will pass the user’s request to the sensor component of access bank account page. The sensor component will check the permission code for page and then will send the permission code to MDM. Also, the sensor component will send a request to CAA database to retrieve all user information required by MDM, such as a valid voting from past request or not. MDM will read the policy rules related to permission code XXX. In addition, MDM will read how many valid votes are available. MDM, based on that information, MDM will decide by either generating probate document for user to get access rights to back account page, waiting to receive more votes, or send request voting to all group members.

For our prototype, we have applied our trustworthiness policy rules that are explained in Chapter 5.
Figure 3-5 CAA component, Sensor Module, and Making Decision Modular MDM, CAA database sever, and APNs server
Chapter 4 Implementation and Testing CAA System

Our prototype is designed and tested on an iPhone. We created a Mobile App for Community-based Authentication and Authorization CAA by using a new language called Swift mobile programming language. Swift Language was created in June of 2014. The reason using Swift Language is, an app on IOS and OS X is trusted by some people because each single app on Apple store is signed and provisioned by Apple to download and install to any device [18]. There are strict Apple approval measures for each app submitted to Apple store. Secondly, we believe the next generation of applications will be in mobile apps. Also, we believe that the next generation of programming language will be in mobile programming language. Thirdly, through our observation, we find an iPhone the most popular smart phone in use. Lastly, the growth of using mobile phones can help us to improve the usability of using CAA systems. CAA system requires a quick response from community group members and by using an app on a mobile phone, response time will be shortened since people take their phone wherever they go, compared to desktop applications or websites that tend to be static in location.

In initial step, our prototype CAA app, called \textit{mfauth} can be deployed on both user and community group member devices. A user can utilize a CAA app to register or log in. Every user needs to register on our system by entering correct credential information to
log in, such as username/password, first name, last name, E-mail, and his phone number. Then the user should select a group of known and trusted people who will help the user identify himself to the system by voting in real time. The user’s group members must register their mobile device by using Register friend’s device. Register friends device is used to send a Device Token of group member to a remote server in turn sending a notification message to enable their voting. The group of members should be trusted people who agree to help. The group members as a whole are working as one factor for multi-factor authentications. Every user can find at least one trusted group member around him. In fact, we are living in group. For example, the user and his wife are a group, the user and his parents a group, or user and his classmate a group, or roommates are a group. The user should select a trust group who trusts them and offers to help him as members of his trust group. The group should be available and willing to help each other in term of authentication.

A user can enter group members either manually or by import from Facebook. We prefer a user to import group members from Facebook to save time and to provide a large knowledge base about both the user and his friends to our system. All data will be sent to a very secure and trusted remote system to protect data privacy. How we can keep a remote server system secure and keep data private is out of our scope and we will discuss it in further work.
In login phase, a user must provide correct credential information, Username and password. Passively, the user’s current location will be sent with his credential information to a CAA server. The CAA server will process and validate the Username/password and to be ready to send SMS and APNs-OPT to the user’s group members. If the credential information is verified, SMS and APNs-OPT will be sent to user’s group members. The CAA system will inform the user if all SMS is successfully sent. A link is indirect URL to CAA server to assist identifying the user’s validity by asking group members, *Do you know this person?* To assist in validation, a picture of the user is also exhibited. When the voting page is opened, a message box is displayed to request if community group member will allow the sharing of his/her current location with CAA. This step will allow us to gather two factors from each member of the group,
including current location. Additionally, the verification code sends to the mobile group member’s hardware device. A hardware device is difficult to attack unless it has been stolen or lost. The group will not be able to vote without entering a verification code that should be received by notification message.

Figure 4-3 shows a sequence diagram for connection between a user, APNs, and CAA server after the user provides his/her username/password. In the verification phase, when the system gathered all votes, and current locations of friends, the system will make a decision by either allowing the user to log in or rejecting him. In verification phases, system will see if the group members have voted. If the group members of user have voted, the user will allow the user to log in. Otherwise, the system will either reject the user or allow the user access with restricted access allowing only limited privileges. Restricted access does allow increasign usability in some cases, e.g., access to data that is not sensitive.

In the verification phase, the system will attempt to match the user’s current location with the location of some (or any) of the group members. If there is a match in locations, the system will recognize the group member as trusted which will increase voting weight. Keeping our system simple, we established the rules such that when the system finds at least a few of the group member location is near or matches the user location, the user will be allowed to log in.

It should be mentioned that user and group member location can run the risk of man-in-middle attack, or to eavesdropping attack or modification attack. We can remedy
this issue by applying SSL connection for all connections between the user and server and between the group members and server.

Figure 4-2 A sequence diagram among CAA Server, APNs, and Community group members.

We will discuss our model for trustworthy factors in our multifactor authentication in the next subsection.
Figure 4-3 Mobil App for CAA system V.0. The left picture is a main interface, the middle is registration interface, and the right picture is SSO with Facebook to important information from Facebook.
Chapter 5 Trustworthiness on multifactor authentication with CAA System

In the environment of multi-factor authentication, there are many factors that need to be managed. Some of these factors are considered weak. For example, a user’s location is not unique. However, he/she has multiple locations, each with a different frequently. In CASA[13], the most frequent locations are at home and work. The question is how we can make this a stronger factor. A simple solution is to assign a frequently for each location so that the most frequent locations can be a stronger factor due to a higher possibility of being in those two locations. In CASA[13], they provide a probability model framework enabling the selection of an active factor for mobile based on a passive factor, such as current location. They studied how to select active factor for android devices where some people do not enable a PIN for their cell phone. Also, they believed their model could work for online services as well which was beyond their paper work. The probability model is based Bayes classifier in the following equation[13]:
By substituting the values of the selected factor, i.e. the location factor, we can conclude that [13]:

\[ \hat{u} = \text{sign} \left[ \log \left( \frac{\alpha P(u = 1)}{P(u = -1)} \right) + \sum_{i=1}^{n} \log \left( \frac{P(s_i|u = 1)}{P(s_i|u = -1)} \right) \right] \]

In above equation, the right side is a selected active factor comparing PIN and home current location factors. In their model, a classification model had 92% accuracy which we do not think is good enough for authentication on remote services.

Our model is used a simple design based on a Community factor (somebody knows you). We have six factors on our system. They have had a different weight so that some of them have a higher weight than others. In our system, we will add one vote of community-based factor as a risk assessment to achieve some balance and equivalency between all six factors. Figure 5-1, exhibits our design of a multifactor authentication system based on community-base.
The Community-based factors authenticate based on voting. It has group member information including their voting. When username/password is validated, a APNs-OPT will send to group members. The SMS contains an indirect link to remote server voting page, while the APNs-OPT send by using remote notification to group member’s devices. Group members has two way to voting either by SMS and that will take long time due to delay time for delivering the SMS. the another way to voting by using CGM App which showing at figure 3-3. The user will open the link on SMS and answer the question by Yes or No. Because the remote server needs to guarantee that voting is coming from right person, each group member has to enter the verification code which will be sent to their IPhone device though the notification. With the completion of this step, the Server will obtain two things from group members, a Voting and the current location of group members. Voting is needed verification of a user by displaying to the group member a picture of the user and questions whether or not he/she is known by them. Current location of members is necessary to evaluate who is near the current user location. If there is at
least one trusted person (group members) around the user, such as his wife, the potential legitimacy of the user is increased.

Our assumption here is that all group members need to be active. Any request coming from Community-based factor should respond quickly. Community-based system helps community to identify individuals.

*Factors box* has six factors with different level of weight. A vote that is coming from Community-Based factor from previous phase needs to be as smooth parameter. If any of factors from factors box is matched, the system will allow the user to log in. The factors box has six factors as all group members, current location of user, Friends’ current location, Device Token, One Password Time, SSO with Facebook.

Community-based factor is a factor that depends on community group members CGM. CGM is a group which consists of trust people for user such as user and his wife. As we mentioned previously, CGM can have at least one person and the as maximum of five people. The sequence to log in with this factor is that a user will enter his/her username, and password, then the system will verify the user based on username, and password. If he/she is verified, the system will move authentication to next layer which is Community-Based factor by sending SMS and notification to all CGM. The SMS will contain a link for voting page, and the notification message will contain a verification code which the CGM will use to send his/her vote. Moreover, CGM will also send their location. As soon as one voting for CGM arrives, the system will move the authentication to the factors box. If all members have voted, the system will legitimize the user and allow to him/her to log in.
For user current location, it will be considered as a low weight factor unless current location user is matched with one of the most very frequently location such as home or work location. We keep track of the history of current location user whether the user is legitimate and illegitimate. Every time the user is consider as legitimate user and he/she is successful log in with the system, a server will add to current location credit to next time. In contrast, when a user tries to log into the system from particular location and fails, the current location will decrease credit for this location. Denial-of-service DDoS attack can mess with this feature. For example, user’s work location can be one of the most frequently user’s current location with high possibility to be legitimate user. There is potential DDoS attack is coming from co-worker to prevent a legitimate user to log in with current location matched at work. When attacker attempts to log into user account from work, then every attempt, will decrease the number of credits for user’s work location for legitimate user. Therefore, we can solve potential of DDoS attack, by just keeping track of the current user location when he/she is considered as legitimate user.

Our system will learn about current location of user. The history of most frequent current location can be used for a match with current user location. For example, let’s assume that the system keep track of user’s current locations when the user is logged in; the user always logs in from home and work base on study one[13]. The system will find user’s home location to be the most frequently location of user. Let assume that user tries to log in form his home. CAA system receive one vote which is the smooth parameter either from a user himself or his CGM. Then, the authentication will move to the next layer which is factors box. As soon as the system finds that the current location of user is
matched with the most frequently location, it will allow user to log in because the current location of user has high possibility of being a legitimate user.

Friends’ current location is current location of group members who are sending the votes. In some cases, a user and one or all his/her group members can be in close proximity to each other. For example, when a user, and his wife, who is a group member, hang out together to have a dinner. The current location of user is not from the most frequent location. A restraint is, does the usually a user go there. However, the system will find that the user is close to one of his group members. When the system receives a vote from his wife and the user’s current location is matched with Friends’ current location will increase of possibility of a user is authorize and increase of weight of current location of user.

There is the potential for attack from people around a user. However, in the movie, *Godfather*, Michael Corleone said “Keep your friends close but your enemy closer”. The user should be careful who is around him when he/she wants to log into secure system. Not all people who are near the current user can intervene and send their location to CAA system; instead, only people who received the SMS and Notification message can send their locations to CAA system. SMS and notification message is used to gather current location in such a reliable way, and to ensure the confidentiality and integrity. The feature nearby can be used to find all people who are group members. However, we use the SMS and notification messages to ensure the confidentiality and integrity.

_Device Token per app_ can be a unique factor since it is a unique identifier for iPhone Device. Each app on IPhone devices can have a different unique Device Token.
However, a unique Device Token per app is indicated to one particular device. Device token per app is similar to a phone number per SIM card. SMS can be sent to particular person based on phone number who owns the SIM card, while notification message can be sent to particular app on particular mobile device based on Device Token. How the Device Token is generated and how the notification messages are sent is described in Chapter 2. In the registration phase, all group members have to share their Device Tokens of CAA app on their mobile devices with CAA server to enable the CAA server to send notification message to their mobile devices. Since the unique device token is available on CAA database from registration phase, we can use Device Token identifier of user’s IOS device as factor of user. When user attempted to log in with CAA app from his/her mobile IPhone device, the user can provide the Token passively to CAA server to match with what was stored with his/her information.

There are two kind attacks which can occur, Spoofing attacks and eavesdropping attacks. We have two solutions to solve such attacks. One of them, by sending an acknowledgment notification message through SSL channel to user, and the user can confirm that he/she tried to log in. If User receives server’s acknowledgment, user can acknowledge the server as well, and then matching can be accomplished. That will remedy the potential of modification attack that can happen. For eavesdropping attack, since that Token identifier will be available on both sides from the initial phase, such hashing functions MD5, and SHA1 can be used in both sides for Device Token identifier to remedy eavesdropping attacks.
One Password Time OPT is used based on remote notification is used on our model as we mentioned in Section 1.2.

Signle Sign On SSO with Facebook authentication services is that increasingly used by many websites and applications. SSO allows for users to log into website or applications without creating an account. Instead, users can use services such Facebook and Instagram, etc. to log into a website or application. In our system, we utilize SSO by using Facebook.

The effectiveness of community-based factor depends on how many group members are active and voting to help user. The community-based factor is effected by many factors, such as time of day, lost carrier signal for SMS message. If the user used a community-based factor to log into the system at midnight, the group members may or may not reply. In this case, we have to instead use factors such SSO, current locations of user, E-mail, token device or previously used information by asking questions about it. For password reset, we are using the most a traditional way by sending a temporary E-mail. In addition, the user can add and delete group members whenever he asks for it.

To solve of voting delay due to SMS delivering by carrier, we do another option for all community-based factor by implement CGM app for voting. Al group member can use this app just for voting. The app is be lucked for voting until the APNs that has request for voting arrive. Then the interface for voting will open. See figure for CGM app.
Using OAuth with SSO in multifactor authentications system is required to store more information about each user’s Facebook account who intend to use SSO with our system. Many social network websites offer Single SignOn (SSO) services to other websites to allow users to use their social network account to access their services. A website needs to register its application to social network providers such as Facebook and Google to adopt SSO, and to be an affiliated website with social network sites, and to be a trusted third party. See Figure 5-2, there are many SSO providers. The challenge is validating a user if a Facebook account that is logged in is owned by the same user who tried to log into our system. We solve this challenge by storing some information in the registration step. When the user registered to our system by using his Facebook account to import information, the initial information about his Facebook is stored, which is why we prefer all users to log into their Facebook account in initial steps. The user will trust our system and his Facebook information is encrypted.
5.1 Limitation of Factors in Factors box in Our Model

5.1.1 Location accuracy

In locational terms, latitude and longitude are not very accurate. Measurements are expressed in degrees and the distance between $1^\circ$ and $2^\circ$ is approximately 70 miles. Therefore, the most accurate way to apply latitude and longitude locations is to express them in minutes, there being 60 in each degree, and seconds, 60 seconds in each minute. Theoretically, one minute of degree is approximately 1.1 miles and one second is about 100 feet. What this means to us is when we attempt to achieve accuracy of location using degree seconds there is the potential for location within 100 feet, not accurate enough for our use.
In practical usage, we attempted to use Geolocation for HTML 5 to gather current location of friends and Swift Language’s CoreLocation to obtain the current user location. The accuracy obtained was not acceptable. Both Geolocation and CoreLocation do not provide the accuracy we needed, although CoreLocation does seem to provide slightly better accuracy than GeoLocation based on our gathered data.

5.1.2 Limitation of Sending SMS by Carrier

Delay time for sending SMS to all group members is unacceptably beyond our limitations due to third party usage. We discovered that the voting time is sometimes delayed up to or more than ten minutes due to group members not receiving SMS at login. That delay effects our final results. This problem is related to the user’s mobile device carrier. In our testing, we sent approximately 138 SMS messages and, yet, one group member did not receive even one message.
Chapter 6 Performance Evaluation of CAA

In Chapter 6, we display the results of our proposed design. We applied several evaluation metrics in our proposed system: Correctness, Trustworthiness of authentication, and performance measurements, including execution time and storage requirements.

The prototype utilized a Community-based factor for user authentication by using group members. We applied a Swift Language, a new mobile programming language. Swift Language is founded on WWDC 2014 when Swift 1 was released. Our prototype applied Swift 1.2. Our mobile app used Xcode 6.1 beta version and tested on the IPhone 5S 32Gb IOS 8.1.

We created an Apple developer account, then requesting identity authority in terms of enabling us to test our model on a real IPhone device instead of using an Xcode simulator. We created an app, id mfauth, for our prototype and enabled Push notification service for the mobile app. Push notification service is necessary for remote notification, in particular, the APNs-OPT (Apple Push Notification Service for One-Time Password) service provided by our system.
To configure push notification for our iOS App mfauth, a Client SSL Certificate was generated to enable our remote server, a Community-based server, to connect with APNs server in an SSL connection to send notification messages to the user’s mobile device. The APNs requires a unique Client SSL Certificate for each Apple ID.

For SMS sending, we utilized PHP Mail function `mail('PhoneNumber@tmobile.net', 'Subject Of SMS', 'Contain of SMS'),` where PhoneNumber@tmobile.net is an E-mail to text service from a T-Mobile Communication provider. PhoneNumber is a destination of SMS mobile number. It must be replaced by any T-Mobile mobile number.

To integrate SSO (Signle Sign On) with Facebook for validating users and obtain profile information about their friends, we wrote code that utilizes Facebook IOS SDK 4.2 with Graph API version v2.3. We requested a Facebook developer account and entered the Facebook ID app, Facebook name, and CFBundleURLSchemes to info.plist of our Xcode project.

For our community authentication server, we set up our environment running Linux 2.6.32.59 OS with Apache web server and Percona database server on our private domain on Siteground.com. We purchased the domain for $100.00. Our Linux server environment has the following configurations:
### Table 1 Software and hardware Specification for our experiment preformed

<table>
<thead>
<tr>
<th>SERVER INFORMATION</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Server OS</td>
<td>Linux 2.6.32.59</td>
</tr>
<tr>
<td>Web server</td>
<td>Apache 1.3</td>
</tr>
<tr>
<td>RAM Size</td>
<td>10GB</td>
</tr>
<tr>
<td>CPU speed</td>
<td>2000.025 MHz</td>
</tr>
<tr>
<td>PHP version</td>
<td>5.6.12</td>
</tr>
<tr>
<td>Database Server type</td>
<td>Percona Server</td>
</tr>
<tr>
<td>Database Server version</td>
<td>5.5.44-37.3-log - Percona Server</td>
</tr>
</tbody>
</table>

### 6.1 Performance of CAA system

In this thesis, we conducted experiments for the performance comparisons of different techniques and technologies including methods of delivering the OPT such as SMS, Email, and APNs, and also CAA compared to the traditional authentication and APNs-OPT in order of login.

#### 6.1.1 Compare among SMS-OPT, Email-OPT, and APNs-OPT.

In this thesis, we discuss different schemes for transferring the OPT to users in order of use for user authentication. We discuss the different schemes in more detail in Section 1.2 and in Chapter 2.

SMS-OPT is a method of using OPTs over a cellular network. In our experiments to evaluate the performance of SMS-OPT, with the help of a group of 10 researchers in
our university, we sent SMS to each of them at the same time. Most of our participants use T-Mobile carrier.

E-mail-OPT is a method of sending OPTs to user’s E-mail instead of using SMS. in our experiments to evaluate the performance of E-mail-OPT with the assistance of a group of 10 researchers in our university. We sent E-mail to each of them at the same time. Most of our participants use gmail.com.

APNs-OPT is a method of sending OPTs through an SSL channel through an APNs server hosted by Apple. In order to use APNs-OPT, a user has to use Apple products such as IPhone, IPad, etc. in our experiments to evaluate the performance of APNs-OPT with the assistance of a group of 10 researchers in our university. We sent APNs-OPT to them all at the same time.

Figure 6-1 displays the comparison performance results for the different schemes, SMS-OPT, E-mail-OPT, and APNs-OPT. The executing time is calculated in terms of user login to the system using OPT. We found that APNs-OPT provides the best performance when compared to the other schemes, although sending messages through APNs takes more initial time than others schemes due to pre-log in SSL handshake. However, SMS-OPT and E-mail-Opt suffer delivery delay.
6.1.2 Performance of CAA system

Community-based factor requires interaction of group members and requires a quick response from them to a system. As quick response will improve the usability of our design, we performed a study to see how quickly group members can respond using different methods including SMS, remote notification messages, and E-mail messages. We actually had 16 participants but 6 of them used Android phones and therefore cannot be notified with APNs (remote notification). Our results found the APNs and SMS systems received rapid response while the E-mail responses took more time. APNs outperform SMS with 60% response rate reached in one minute. SMS trial required twenty-three minutes to reach 60%. After 152 minutes (2 and a half hours) the response rate over E-
mail had only reached 18%. We believe that APNs is better than SMS due to message delivery delay for SMS which can happen in attempting to deliver a message to receivers. APNs used Wi-Fi connection to deliver a message. The lowest rate was for E-mail, most probably due to most people checking their email periodically. Figure 6-2 displays the result of our experiments.

We also tested our proposal and compared it with other authentication methods, such as Traditional authentication, OPT over APNs, and Our community based-factor. Our community based-factor has a user has a user group of three people. Community base-factor tested based on the group members responding as quickly as they can. For our system, if a user used only username/password to log in, he/she will log in with average time 194.93 millisecond while OPT over APNs takes around 8131.18 milliseconds. The minimum time for a user with three group members to log into the CAA system needs to be 71955.88 milliseconds. See Figure 6-3.
6.2 Trustworthiness of the authentication

In our system, we used the Community-based factor authentication as a factor for multifactor authentication. In term of increase sign the trustworthiness, reliability, and availability of our system, we integrated with Community-based factor authentication six factors (username/password, current location for users, current location for friends, Token of device, Facebook social network).

We categorized them as a two factor group each by including one vote from group members as a credit for each category. Thus, the total will return three factors for each group. We can classify our system as community-based authentication assuming three factor system authentications. When all group members have voted, it is considered credit score if we receive three matched factors.
Over time, the system gathers information about the user and his friends, increasing knowledge about the user and his group members which assists in the user authentication thereafter. It also helps to weight group members by whom is active to voting. Moreover, we can identify friend’s members by their location. For example, if a user requests a voting to log in from his group members, the voting for one member may be received from a familiar location such as his home. Therefore, we can trust that vote and perhaps increase the weight of his vote as double credits. In addition, we can use the same information of all group members for their login thus providing a sharing benefit for all group members who share in voting.

When all group members are registered for our system and the system recognizes them as group members, it could use the history of current locations for each group member as a history of current location of all group members. That will help for some cases when no one responds for voting which does not provide us with either or and locations from non-responding group members.

In emergency cases, we enable SMS and E-mail for user when he needs to reset the password. By sending a link for user either on his phone or E-mail, it will be easier for him to reset his password.

A user can change his group members at any time. Any member of a group can be removed by user and the member will be informed via SMS, “you have been removed from a group membership”. In terms of removing a group member, user needs to verify himself through SMS and E-mail for security reason to prevent an attack to remove group members.
Our system will keep a voting live or valid for up to 24 hours, following which, the system will request a new voting. The next graph shows the comparison of authentication models.

### 6.2.1 Storage Requirements of Authentication Methods

We used **Instruments** tool, which part of Xcode tools, for testing and analyzing our app in term of the performance and storage on real time. **Instruments** tool results show that our app take 19.80 MB of real memory at real time. Instruments allow a developer to collect performance data and app behavior over a mobile device during process time. Instruments provide a variety of graphs to display gathered data.

![Real Memory Usage](image)

*Figure 6-4 Storage needed by our CAA mobile app on IPhone device.*

### 6.2.2 History of Current Location of User and Friends:
Based on history of current user and friends location, there is bound to be matching data. The matching happened for the first fourth fraction digits for latitude and longitude. A user and his friends hang out together somewhere that they frequently visit. Based on history of current user location, the most frequently location is either for home and work. The remaining history location is matched with at least one friend location. In general, we know that most of user spend time at home or work, the remainder of their time is when they associate with friends, wife, son, somewhere often frequented (at restaurants or doing exercise, etc.).

Our study on history current location of friends for our experiments showed that some of our participants have sharing the same location at voting time. If those participants paly role of user and group members in CAA, a user’s current location will match with current location of friends. That will give credit as trustworthy for this current location of user in authentication. See figure 6-4.
In our system, each location matching will be given credit for user to log into our system. Our system, a community based multifactor authentication system, is based upon group members and factor authentication. At least three factors must be matched to allow user login.
Chapter 7 Lessons Learned

During the completion of this thesis, I learned many valuable lessons, an integral part of any research work. Reading about and investigating a many-dimensioned topic requires hard work to be able to fully understand the topic. It was necessary for me to explore and examine a vast amount of research papers, documentation, and ideas and, from that, assimilate a wide variety of loosely arranged information before I could begin to develop my thesis ideas which, in turn, became my personal contribution for this thesis. Along this route, I have learned many things about Multifactor Factor Authentication, Cyber-Security, social network data, and IOS mobile. Indeed, I was required to spend much effort to learn the new Swift programming language for iOS devices which allowed me to implement and test my CAA apps.

7.1 Research Experiences

I began my research by reading and investigating about authentication systems in many respected documents and research areas, such as authentication in computer security, authentication in cypher-security, architecture of authentication, definition of one-way authentication, definition of two-way authentication, biometric in authentication, and authentication mechanisms. I learned that security issues exist in all of one-way authentication, two-way authentication, and biometrics in authentication systems. One-way authentication is a traditional authentication which depends on something we know,
and it suffers from many security issues such as disclosure, forgotten data, sniffing, phishing, keylogger and guessign by human or by computer algorithms. Biometrics in authentication is the effective and accurate method for identifying individuals. However, authentication by using biometrics does have security issues when the biometric data is compromised. It is difficult to recover data and/or a system. Authentication mechanisms are of many forms, such as hardware mechanism (hard Token, soft token), software mechanisms (generate token software, third party software, etc.), and encrypted mechanism (Public key encryption and Symmetric key encryption), mobile mechanism (SMS, OPT, etc.), location mechanism, etc.

I discovered a new future research area for me in mobile cloud computing which has many inherent challenges. The main challenge of mobile computing is restriction and limitations in term of capability of CPU, memory, network, etc.

7.2 Build a New Skills to Future (Swift Language)

By doing this research, I developed new tools for future use, either as a researcher or in a career path. In 2014, Apple released a new language, Swift programming language, which it started as was a powerful language. Following are some reasons that learning this language has been so appealing to me:

As career, based on [19], an average salary in the U.S. for IOS apps developer is $101,000 per a year. Recently, most enterprises, organizations, Universities, banks, etc.
own apps or tried to develop their own app. Most of them provide a job position as mobile developer. Based on “apps developer” sold 10 million iPhone 6 in the first 3 days following release. These number forecasts the future direction of mobile app programming. Moreover, WhatsUp app is one of the most popular apps for exchanging free messaging between people around the world. In 2013, it sold to Facebook for approximately $21.8 billion. Based on “businessinsider.com”, Snapshot is going to generate $50 million in revenue in 2015. I think the future program career will be for mobile programming language.

In academic thought, I think mobile programming language will be the future programming language and should become a core course in computer science programs.

In research field, mobile computing and mobile cloud computing is considered as a new research area in which not many researchers are currently employed. That provides a greater opportunity to publish in the beginning. As a researcher, it is necessary to have mastered at least one mobile programming language to demonstration my model.

### 7.3 Discover many knowledge about multifactor authentications

There are hundreds of models proposed for authentication in one-way authentication, two-way authentication, and biometric authentication. During my research and exploring deeply inside all those authentication types, I discovered many factors can be utilized for authentication. That provided me with the ability to evaluate security for
these factors, define a threat model for multifactor authentication system, and determine how we can remedy this security threat.

### 7.4 Challenge of Mobile Computing

Mobile computing has many limitations and restrictions, such as capability of CPU, Memory size, and network connection, not sufficient power, security risks and heterogeneousness of network. All the pervious factors should dealt with carefully when working in mobile computing as developers, researchers, and owners. In contrast, mobile computing has many benefits, such as ease of access to services either on mobile or over network, entertainment, cloud computing, increasign incoming, social goals, etc.
Chapter 8  Future Work

8.1  Improve the CAA Server Components

Making Decision Modular MDM is a core of the CAA server component at the CAA system. MDM is a module that applies policy rules to trustworthiness factors on CAAS. We applied simple policy rules to trustworthiness factors. However, we need to improve the policy rules to cover many contexts.

8.2  Framework to test mobile app based on app category

Instrument is a good tool to analyze the performance of apps IOS. However, it may not give permission compatible with your app or with another app which is of the same category. For example, all banks own certain type of apps. We need to create a framework to compare performance for other apps in same category

8.3  Machine learning and data mining to ask question from social network
Today, machine learning and data mining solve many problems. One of problems that faced us in this thesis is how to prove that a Facebook account belongs to a given person. One of the ideas we had was to use machine learning and data mining to analyze a user’s Facebook profile in order to create a very critical question to ask for user verification.

8.4 Virtualization to Simulate Mobile Devices

In the last decade, mobile computing has drawn the attention of many researchers. These researchers had developed their model and they needed a good environment in which to test their model. One solution would be a virtual framework which is necessary to help researchers model their prototypes and designs online.
Chapter 9 Conclusion

We designed and implemented a Community-Based Authentication and Authorization (CAA) for its users to pre-designate community group of members for authenticating each other. As soon as a server receives an access request from the user, it sends an SMS or utilizes Apple Push Notification service (APNs) to a pre-designated community group of members, requests them to authenticate that user and/or to approve critical access of valuable shared resources. The preliminary evaluation of the prototype shows that with the 60% threshold of 10 willing community members, the approval can reached within one minute.

Based on our experiments, we learned that APNs-OPT compared to other methods of delivering OPT or message, such as SMS-OPT and E-mail-OPT, is much faster. Although, APNs suffers of high latency due to SSL handshake in the beginning of the authentication process. It still has a lower latency time to log in. SMS-OPT and E-mail-OPT suffer a longer delay to deliver a message.

Prior unfavorable conclusion for using CBF in Authentication need to be re-evaluated, since with the new responsive mobile cloud system, the usability of CAAS with CBF can be improved. Our experiments show that responding rate of group members with CAA’s sent APNs message, the best response rate was 60% of group members responding within one minute. The worst response rate was 19% within 17 minutes.
To improve response rate per minute, we integrated six factors with Community-based factor. The six factors are Community-based factor, Current user location, current friend location, one time password APNs-OPT, and Single Sign-On with Facebook. We compute trustworthiness as a three-factor authentication based on Community-based Factor.

Finally, we believe that CAAS can possibly be used to implement in real time since we received a response rate of 60% per minute. Also, we believe that there are many applications that can utilize CAAS architecture, including Inheritance granting or referral. In this thesis, we just realized a simple prototype. We plan to conduct further research for improving its features and performance.
References


[7] “intant Ciphertext only cryptanalysis of GSM ecncypted communication.pdf.”


Appendix A

Set up and Test Manual:

A.1 Installing Web Server:

We purchased a domain ocaprojects700.com to be our remote authentication server which supports SSL connection, PHP 5.4 script server side, mysql database, and mail’s php function to send SMS. The remote server is reachable from everywhere.

A.2 Building The Database:

1) By going to phpMyAdmin page, you will be asked to create your database. You can create one with name “”. Under “ocaproje_Alkhattabi” database you can create the following tables:

```sql
CREATE TABLE LoginTable ( Username VARCHAR(30) NOT NULL , Password VARCHAR(20) NOT NULL, PRIMARY KEY (Username), Phone VARCHAR(30) NOT NULL);
```
CREATE TABLE UserINFO (fname VARCHAR(30) NOT NULL, Username VARCHAR(30) NOT NULL, lname VARCHAR(30) NOT NULL, E-mail VARCHAR(30) NOT NULL, Address VARCHAR(200) NOT NULL, Latitude REAL NULL, Longitude REAL NULL, Phone VARCHAR(30) NOT NULL, FacebookID VARCHAR(100) NULL, AddedDate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP , PRIMARY KEY (Username))

CREATE TABLE RelationsBetweenUsersAndFriends(Username VARCHAR(30) NOT NULL, fphone VARCHAR(30) NOT NULL, FOREIGN KEY (Username) REFERENCES UserINFO(Username), FOREIGN KEY (fphone) REFERENCES FriendsTable(fphone))

CREATE TABLE UserHistoryLocation (Username VARCHAR(30) NOT NULL, Latitude REAL NULL, Longitude REAL NULL, AddedDate TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP, FOREIGN KEY (Username) REFERENCES UserINFO(Username))

CREATE TABLE FriendsHistoryLocation (fphone VARCHAR(30) NOT NULL, Latitude REAL NULL, Longitude REAL NULL, FOREIGN KEY (fphone) REFERENCES FriendsTable(fphone))

CREATE TABLE RequestTable (Id INT(11) NOT NULL, Username VARCHAR(30) NOT NULL, fphone VARCHAR(15) NOT NULL, statue VARCHAR(30) NULL, Notifited VARCHAR(10) NOT NULL, Date TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP, FOREIGN KEY (Username) REFERENCES UserINFO(Username))
CREATE TABLE `ocaproje_root`.`weightFactor` (  
    `Username` VARCHAR( 100 ) NOT NULL ,  
    `Username-Password` INT NULL ,  
    `Votes` INT NULL ,  
    `Oauth` INT NULL ,  
    `DeviceToken` VARCHAR( 100 ) NULL ,  
    `Current_location` INT NULL ,  
    `Friends_location` INT NULL ,  
    `tags_on_Facebook` INT NULL  
) ENGINE =InnoDB ;

CREATE TABLE facebookFriendsINFO(FacebookID VARCHAR(100) NULL, Latitude VARCHAR(13) NULL, Longitude VARCHAR(13) NULL, ID1 VARCHAR(30) NULL, Name1 VARCHAR(30) NULL, ID2 VARCHAR(30) NULL, Name2 VARCHAR(30) NULL, ID3 VARCHAR(30) NULL, Name3 VARCHAR(30) NULL, ID4 VARCHAR(30) NULL, Name4 VARCHAR(30) NULL, ID5 VARCHAR(30) NULL, Name5 VARCHAR(30) NULL, ID6 VARCHAR(30) NULL, Name6 VARCHAR(30) NULL, ID7 VARCHAR(30) NULL, Name7 VARCHAR(30) NULL, link longtext NULL)

A3. PHP API

- PHP class to connection with Database : 

Our PHP class to connect to our database:

[http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip](http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip)
A php fill called “SignUpInfo_form.php”:

A PHP file which is used to register Phase. It has a HTML 5 and PHP codes downloadable from http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip

A php file called “Authenticator1.php”:

A Phd file is used when the user tries to log in. It will obtain information which is sent by Swift language on a mobile phone. Authenticator1.php has many tasks:

1. Initial authentication to see whether or not a user has an account.

2. Inform a user about any events that occur on CAA server during user authentication, such as result of MDM modular decision.

3. Checking for a valid voting from past requests.

You can download it from:
http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip

A php file called “Authenticator2.php”

A PHP file has responsibility of gathering voting information and validating the OPT from group members such as gathering the OPT from group members and compare it with what is on the server. Also, it has the responsibility of storing gathered information to CAA Database, and notifying MDM to recompute a new decision to notify Authenticator 1.
You can download it from
“http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip”

➤ A php file called “UserLocation.php”

A file which is used to store current user to UserHistoryLocation Table when a user successfully logs in.

You can download it from:
“http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip”

➤ Javascript file called “GeolocationFile.js”:

Used to get group member current locations.

You can download it from:
http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip

➤ PHP file called “RegisterDevice.php”:

A PHP file used to send a Token user device to CAA Database. Token Device is required in order to send the OPT to user through APNs.

You can download it from
“http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip”
PHP file called “OptOverNotification.php”:

A PHP file is used to verify the OPT that is sent to the user by APNs along with what has been stored in CAA database. Moreover, it can be used to confirm any changes, such as password changing, Token Device changing.

You can download it from:
http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/MfauthServer.zip

A4. Swift Language:

Swift Language is a new language for IOS and OS X Apps. The language was created in Jun 2014. The version 2 of Swift Language was released Jun 14, 2015. When Apple introduced Swift, they talked about three key considerations which is being safe, modern, and powerful. In terms of develop an app by using the Swift Language, you will need to have a Mac with the latest version of OS X. Also, you need to download the latest version of Xcode such Version 6.1.1 or earlier.

In general, Swift Language, like any other language, has its own vocabulary and grammar (syntax). It has variables, loops, arrays, and more. Swift Language is completely an object-oriented language. A developer can utilize this language to gain access to Cocoa and Cocoa touch frameworks. It can also work side-by-side with Object C by using bridge file.
Apple still evolves and updates Swift Language and update it. Swift Language is just getting started as a powerful language and I believe it will be the future language for mobile IOS app. There is not many resources to learn this language except a few such as:

A new book entitled “The Swift Programming Language” by Apple Company

Here is some links that I found helpful:


2) Stanford University developing iOS 8 Swift Apps

Www.youtube.com/watch?v=XXhcJKKDOtI

You can download all Swift Language projects on this “http://walrus.uccs.edu/~gsc/pub/master/kalkhatt/src/mfauthXcodeV7B5.zip”

To create project for Swift Language, you can follow the next steps:

➢ Go to Xcode on your Mac, and open it.

➢ From file select new -> projects
Choose a template for your project. From left side select *IOS -> Application* and on right side select Single *View Application*.

You need to enter information which is important when you need to distribute your app. It is necessary to select type of language to Swift.
Figure A. 3 Product Name and Bundle identifier for iOS APP

- After that you need to save your project in your selected place

- Now, you are ready to create your project.

Figure B. 1 Explore Xcode's User Interface
A5. Facebook SDK with Swift:

Facebook SDK for IOS is a framework, which helps developers to integrate Facebook websites with your app. There are advantages for both users and developers. Using Facebook SDK for IOS, allows users to connect to their Facebook account in terms of post and share information to/with Facebook. Facebook SDK allow to users to log into app by using Facebook login as a fast and convenience way. Facebook implements their API based on the OAuth protocol to extent of user experience of social sign-on and social sharing.

In terms of integrating Facebook with your app, you will need to register your app with Facebook by creating App ID with Facebook. You can go to this link:

https://developers.facebook.com/?advanced_app_create=true

Figure A. 4 Apple ID and App Secret for Facebook App

After you create your App ID, you will get the next picture which shows your App ID.
The next steps, link your app with this App Id, by doing following:

1. Download and Install the Facebook SDK for iOS from
   https://developers.facebook.com/resources/facebook-ios-sdk-current.pkg

2. You will find facebook sdk folder on ~Documents/FacebookSDK

3. Open the folder then drag FBSDKCoreKit.Framework, FBSDKLoginKit.framework, FBSDKShareKit.framework to your Xcode Project on your Mac.

4. On Xcode project, under supporting files you will find a file called “info.plist”
By click on “info.plist”, you will get:

Now, you need to add three attributes and values. You get these values when your register. See Figure A 5
Figure A. 8 Three attributes are needed to create in info.plist.

Supply us with your Bundle Identifier

Find your bundle identifier in your Xcode Project’s iOS Application Target.

Bundle Identifier

Find your bundle identifier in your Xcode Project’s iOS Application

Figure A. 9 To add bundle Identifier for your App.

Last step is you have to add your bundle identifier for your app to Facebook developer account. You can go to facebook.developer.com, then go to you’re my Apps. After that you will ask you to add your bundle identifier.

By doing all the steps above, you will link your Xcode project with Facebook SDK.
Next steps, how to using Oauth with swift to connect with Facebook API. All Facebook SDK for IOS tutorials that are on developer.facebook.com used Object C language. As we said the good thing about swift is can work side-to-side with object-C. so, before to connect with Facebook API, we need to create bridge file in our Xcode project.

Create file called “bridge-header.h” as you can see in the next Figure:

![Image](image.jpg)

Figure A. 10 Bridge-Header between Object-C and Swift Languages.
Create the next code on it:

Now, you need to tell swift compiler about this bridge header file. By clicking on the main folder of your Xcode project from Xcode interface.

In the right side, click on build settings, then find `swift compiler – code generation` section. Under object-C bridging header attribute, write you’re the path for your bridge file.
The next steps, go to your Xcode project, find the file called "AppDelegate.swift", then write those two functions:

```swift
func application(application: UIApplication, 
didFinishLaunchingWithOptions launchOptions: [NSObject : AnyObject]?) -> Bool {
    FBSDKApplicationDelegate.sharedInstance().application(
        application, didFinishLaunchingWithOptions: launchOptions)
}

func application(application: UIApplication, openURL url: NSURL, sourceApplication: NSString?, annotation: AnyObject?) -> Bool {
    return FBSDKApplicationDelegate.sharedInstance().application(
        application, openURL: url, sourceApplication: sourceApplication, 
        annotation: annotation)
}
```

Then go to the file called "ViewController.Swift" or any Controller view you want to integrate with Facebook. Inside any ViewController there is main function
called “viewDidLoad()”, then copy and paste the following code to body of this function:

```swift
if (FBSDKAccessToken.currentAccessToken() != nil){
    let loginbutton : FBSDKLoginButton = FBSDKLoginButton ()
    loginbutton.center = self.view.center

    loginbutton.publishPermissions = [
        "public_profile",
        "E-mail", "user_friends", "read_custom_friendlists",
        "user_likes", "user_photos", "read_stream",
        "user_groups", "user_location"
    ]

    loginbutton.delegate = self
    view.addSubview(loginbutton)

    GetGraphData()
}
else {
    let loginbutton : FBSDKLoginButton = FBSDKLoginButton ()

    loginbutton.frame.origin = CGPoint(x: 100, y: 20)

    loginbutton.readPermissions = [
        "public_profile", "E-mail",
        "user_friends", "read_custom_friendlists",
        "user_likes", "user_photos", "read_stream",
        "user_groups", "user_location"
    ]
    loginbutton.delegate = self
    view.addSubview(loginbutton)
    GetGraphData()
}
```

Then create function called “GetGraphData()”, and copy/paste the next code.
In the name of class and you have to add delegate method by adding `FBSDKLoginButtonDelegate`

```swift
class FacebookViewController: UIViewController, FBSDKLoginButtonDelegate {

    var str = "me/feed"

    let graphrequest: FBSDKGraphRequest = FBSDKGraphRequest(graphPath: str, parameters: nil)

    graphrequest.startWithCompletionHandler({(connection: FBSDKGraphRequestConnection!, result, error: NSError!) -> Void in

    })

}
```

Now, Create a two function
Now, you compete the totals of ingrate Facebook API with your app by using Swift Language.

I will upload all my Swift Language file to the “link”
Appendix B

Appendix B describes a demo and how to use my model in user view.

In term of deploying any app on an actual iPhone, you need to request certificates. We have to generate and request two kinds of certificates: a developer and a distribution certificate. We just need to generate and request them once per Apple developer ID Account. A developer certificate is used only for testing your app on your XCode simulator or your IPhone while a distribution certificate is used for publication apps on Apple store and make them available for every once.

A “mfauth” is our community base-factor authentication app which easily can be installed on any IOS mobile. For testing mfauth app, you need the following:

1.1 MAC Prof laptop has the latest of OSX version 10.X.X

1.2 Xcode 6.X

1.3 Apple developer Account

1.4 Register your IPhone at Apple developer account

1.5 USB wire for Iphone
You should link your Xcode with your apple ID developer Account in term of testing your app on Xcode simulator. You can do that by launching Xcode application on your Mac -> Go to Xcode menu -> Click on Preferences -> Click on Add Account -> Enter your Apple ID and Password. See the Figure B.1.

![Figure B. 2 Adding App ID Account in XCode Preferences.](image)

In addition, you need to request Apple development certificate to be able to test your app on your Xcode simulator. After you generate and sign a certificate, you can import it to your Xcode by launching Xcode application -> Go to Xcode Menu on the top -> Preferences -> add Apple IDs -> View Details -> where you will find all existing certificates and provisioning profiles. Now you can click on add signing identities then you can import the signed certificate. See Figure B.2, and Figure B.3.
Figure B. 3 Adding Apple ID Account in XCode Preferences.

Figure B. 4 Existing certificates and provisioning profiles
The next step, you can download our Swift Project from this “walrus.uccs.edu/~kalkhatt/cs700/src” and connect your IPhone with your Mac Pro by using USB Cable. Finally, you can deploy your app to the device by selecting your device as deployable targets, and then run your project to install the app to your device. See Figure B.4 and Figure B.5

After clicking on the *mfauth* icons, the app will be lunching and the initial interface will be shown. If you are a new user, you need to a register by clicking on *register* button. In Figure B.6, CFA initial interface is shown.
By clicking on register button, a registration interface will be shown as in Figure B.7. The user should enter all necessary information which includes (Username, password, phone number, etc.). At bottom of registration interface, there are two chooses for a user to enter his group members either sign a manual way by clicking on “Add More Friends” or import them from Facebook Account. In term to add group members, three kind of information is needed. Every user has to have group members. For each a member, First name, last name, and phone numbers is required to entered and stored to the remote server.
Figure B. 7 Registration Interface need User's information, and a group members information.

Figure B.8 and Figure B.9 are shown Add More Friends manually interface and import Facebook interface friends to select from your friends on Facebook respectively.
Figure B. 8 Add More Friends Interface

Figure B. 9 Import From Facebook Interface.
After a user is registered, he can log in by using his Username and Password by using interface in Figure B.6. As soon as a username and password is verified by a remote server, a SMS will send to all a group members to vote by either verified or refute a user. In Figure B.10, and Figure B.11 are shown a SMS and voting page content.

![Figure B. 10 SMS contents which will send to all a group members](image1)

![Figure B. 11 Voting page contents. The picture is imported from Facebook](image2)
There are three statutes for a user as full access to resource, restricting access, or reject.