A Mobile Device based Identity Validation System for Online Social Networks

Yiwei Shi

School of Computer Science
McGill University
Montreal, Canada

March 2012

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

Abstract

Currently, online social networks (OSNs) do not provide validation mechanisms to verify the identity of a user who is seeking linkage with another user. This shortfall is exploited by attackers to infiltrate other people's social circles to gain access to personal data. Therefore, building an identity validation system is necessary for protecting the user interest as well as enhancing the user experience.

In this thesis I present an identity validation system—CredFinder for OSNs using commodity mobile devices. Three validation protocols are designed under different scenarios people may encounter. Targeted on Facebook, we propose an Android based prototypical implementation including three subsystems, the mobile device application, the validation server and the OSN application server. The implementation results demonstrate that CredFinder is capable of performing identity validation. To the best of our knowledge, CredFinder is the first mobile device based practical system against social network identity theft attacks. The validation strategy in our system gives users the power to connect their online and offline social networks together.

Résumé

Actuellement, l'expérience utilisateur des réseaux sociaux en ligne (Online Social Networks) est difficile lorsqu'un utilisateur a à valider des identités d'autres pour se connecter. Compte tenu des attaques rampantes de vol d'identité sur OSN, il est parfois difficile de distinguer si la personne à connecter sur OSN est celle qui l'utilisateur connaît-il dans la vie réelle. Pour cette raison, la construction d'un système de validation d'identité est nécessaire pour protéger l'intérêt des utilisateurs ainsi que pour améliorer de l'expérience utilisateur.

Dans cette thèse, nous présentons un système de validation d'identité - CredFinder pour OSN développé en plate-forme des dispositifs mobiles. Trois protocoles de validation sont conçus pour faire face à différents scénarios que d'utilisateurs peuvent-ils rencontrer. Concernant Facebook, nous proposons une implémentation basée sur Android prototype composée par trois sous-systèmes: application de dispositif mobile, serveur de validation et serveur d'application OSN. Les résultats et les analyses de l'implémentation démontrent que CredFinder est à la fois efficace et efficiente pour accomplir la validation d'identité. Au meilleur de nos connaissances, CredFinder est le premier système réel basé sur appareil mobile contre les attaques de vol l'identité sur OSN. La stratégie de validation dans notre système donne aux utilisateurs ordinaires le pouvoir de connecter sur leurs réseaux sociaux en ligne et hors ligne.

Acknowledgments

I am fortunate to have Prof. Muthucumaru Maheswaran as my thesis supervisor. I would like to thank him for his continuous guidance and supervision on my thesis research. His advice and suggestions are invaluable for this project.

Thanks should be given to everyone in the advanced network research lab (ANRL) for their valuable comments and discussions. Special thanks to Wesley Ellis and Cong Yu, who provided lots of help on the programming and debugging of my program. I owe many thanks to Jianxun Dang, Kang Wang and Lixiong Chen, who gave me plenty of advice and support during my study of over two years.

Furthermore, I would like to acknowledge the assistance provided by Weizhong Li, Yuan Jin, Dr. Saied Hemati, Prof. Warren Gross, Dr. Camille Leroux, Prof. Xue Liu, Yi Zhang, Yijia Xu and Hanqiang Cheng.

I also would like to express my deepest appreciation to my mother, my father and my girlfriend for their successively spirit support and encouragement all the time.

Contents

List of Figures

Chapter 1

Introduction

An online social network (OSN) [1] is an online service that focuses on building social networks among people, who use these sites to share photographs, update recent status, contact long-lost friends or establish new business relations.

OSNs have been increasingly gaining popularity. Facebook has been reporting over 800 million active users, more than 50% of whom log on to it in any given day [2]. According to Facebook, on average more than 250 million photos are uploaded per day. LinkedIn, the world largest online professional network, claims to have more than 135 million members in over 200 countries and territories as of November 2011 [3]. Other OSNs such as Google+ and Twitter are also being paid a lot of attention. Additionally, the number of users in these OSNs is still growing.

1.1 Motivation

As the influence of OSNs increases, so does their attractiveness for criminals. In 2008, for example, worms primarily targeted users on MySpace and Facebook [4]. Because the intrinsic property of OSNs is users sharing information with others, plenty of personal data are conscientiously and inevitably released in the public domain by the users. This data can be of interest to various entities inside and outside of the OSN, exposing users to different kinds of privacy related threats, OSN variants of traditional network threats, identity related threats and social threats [5]. One of the fastest growing threats among them is the risk of identity theft attacks faced by everybody in the OSN [6].

The key goal of an adversary in an identity theft attack is to obtain personal information

about a victim's friends after successfully forging the victim, and to establish increased levels of trust with the victim's social circle for future deceptions. Typically, an adversary in the first place tries to find ways to obtain a victim's personal information, such as name, location, occupation and friend list from his public profile in OSNs or searching results on the Internet (e.g., Google or Wikipedia). Then, the adversary forges the victim's identity and creates a similar or even identical profile in OSNs. Afterwards, he sends friend requests to the victim's contacts. Once the friend requests are accepted, he builds the victim's friend network and gains access to profiles of the victim's friends. In addition, the adversary can create multiple fake identities related to the single victim and may also forge the identities of the victim's friends [7]. Furthermore, the adversary can also implement an automated, cross-site profile cloning attack [6] in which he can clone the identity of a victim from one OSN where the victim is registered to another OSN where the victim is not registered yet. Having successfully created the fake identity in the OSN where the victim has no account, the adversary can automatically attempt to rebuild the social networks of the victim by contacting her friends who have accounts on both sites. The results presented in [6] show that both of these identity theft attack schemes are effective and adversaries do not raise much suspicion in users who have been compromised.

Most users may be apt to trust people's activities in OSNs more than those on other websites because OSNs are built on the core concept of connecting of people knowing each other. But due to the user's lack of awareness, such trust makes it easier for an adversary to obtain a victim's personal information and then clone the identity. Thus, it is necessary to build a mechanism to detect identity theft attacks and protect the user.

1.2 Thesis contributions

In this thesis, we propose an idea of using people's social relationships in the real world to validate identity for the OSN users. We exploit the fact that the majority of people who want to become "virtual friends" in an online social network probably have met physically before (or at least they have physically met their mutual friends). Implementation-wise, an innovative mobile device based identity validation system for online social networks is presented. Employing the notion "using mobile device as an identity token", the user's social network in daily life can be easily recorded. Targeted on Facebook, we have successfully completed the prototype by developing the Android application, the back-end service and

the Facebook application web server.

Our identity validation system opens up a door to solve the issue that an adversary pretends to be someone in the social network and contacts a victim's friends for malicious purposes. Using people's social network in the real world, we establish a trustful friend network behind OSNs and help users verify persons sending friendship requests. Users transmit their tokens between each other and our system takes the responsibility to connect users' online and offline social networks together. Our mechanism can also be applied to the indirect verification. Moreover, if OSNs take advantage of our system in the process of user registration, many identity theft attackers can be refused admittance at the beginning.

1.3 Thesis organization

The rest of the thesis is organized as follows. Chapter 2 introduces the background such as identity theft attacks and fraud in OSNs. Chapter 3 gives a general design of our identity validation system and Chapter 4 focuses on the concrete implementation of the function for each system component. Chapter 5 shows the performance results and analysis. Chapter 6 states some security concerns of the system. Finally, Chapter 7 concludes the thesis and discusses the future work.

Chapter 2

Background

In this chapter, I give a general description of online social networks and elaborate the online identity especially for OSNs. I then emphasize on different kinds of identity theft attacks particulary those in OSNs.

2.1 Why Social Networking

Online social networks, over the past few years, have rapidly increased in popularity. If you have used an OSN service before, it is likely that you are already fully aware of that. OSNs like Facebook can be thought as your home on the Internet. It's a place people can go to leave you a message, peep what you are doing, or chat with you while you are online. It can be a great way to keep in contact with friends and family. For example, the Facebook "wall" is where it all happens. Based on the core concept of sharing information with others, there are some other popular functions people use frequently in OSNs.

- *Online photo albums and video collections*: In addition to keeping connection with friends and family, an OSN is a good place to upload and share photos and videos. Ordinarily, you'd have to whip out the photo album when people are over at the house, but uploading them to an OSN means grandparents can look at their grandchildren anytime they want.
- *Social games*: There is also a lot more fun in OSNs. You can play online games like FarmVille with your friends whereby you will milk cows and collect eggs from chickens or breed plants and animals.

- *Business*: You can also use OSNs such as Linkedin and Facebook to connect with co-workers and former co-workers to keep the workplace networking going. It is well known that one of the best ways to find a new job is to be referred by a friend or an acquaintance, so keeping up with business contacts is quite important.
- *Finding old friends*: Do you remember your best friend of primary school? With more and more people joining OSNs, they are becoming great places to look up old friends and getting back in touch.
- *OSN applications*: You can also explore thousands of mobile and web applications of OSNs to satisfy various requirements.

The OSN is an astonishing service, however, to enjoy a specific OSN, one needs to register an identity and send "add friend" requests to the people in the OSN, which bring us two crucial issues, online identity and its validation.

2.2 Online Identity

2.2.1 Overview of the Online Identity

An online identity, Internet identity, or Internet persona is an identity that an Internet user establishes in online communities and websites [8]. Rather than use their real names online, most of the Internet users prefer anonymity, identifying themselves by means of pseudonyms, which reveal varying amounts of personally identifiable information.

In the real world, we all have at least one unique fixed identity such as passport or driver license to distinguish ourselves. Its advantage is obviously the security, as long as the authentication process can guarantee that a user is really who she claims to be. However, in the world of Internet, fixed identity is rarely used [9] except the case like that you need to apply a VISA on a government issued website. The downside for online fixed identity is the privacy issue which may cause personal information leak. Chaum [10] argues that fixed identity provides one-sided security, protecting service providers from individual users while the users' information are left in danger. In addition, it is a huge amount of work for the websites to validate whether identities are true or false, let alone the possibility to scare the users off.

A pseudonym is widely used on the Internet. For instance, you can register a Gmail account or QQ (an instance messaging service in China) account using any name that has

not been picked up by others. It gives people a simple and free way to access websites and their services. From the privacy perspective, pseudonyms protect the real identity of its owner. People can present themselves without fear of persecution, whether it is personality traits, behaviors that they are curious about, or the announcement of a real world identity component that has never been announced before. Many people enjoy impersonating with multiple pseudonyms. Someone may like to create one account for her business life and another for social life, for example. Dorian Wiszniewski and Richard Coyne [11] in their contribution to the book "Building Virtual Communities" explore online identity, with emphasis on the concept of "masking" identity. They point out that whenever an individual interacts in a social sphere they portray a mask of their identities. The same thing goes to the Internet and in fact becomes even more pronounced in terms of that an online contributor must make concerning her online profile. She has to answer specific questions about age, gender, address, username and so forth. Furthermore, as a person publishes to the web she adds more and more to her mask in the style of writing, vocabulary, preference and topics. But is it really that easy to hide oneself on the Internet? Recently in China there is a popular term called "human flesh search" referring to search for the real identity behind a pseudonym. It derives from searches that are conducted with help from human users (as opposed to on a purely automated platform, such as Google), often targeted at finding the identity of a human being [12].

It is well acknowledged that a significant issue of pseudonyms is it can be easily abused by adversaries. It provides malicious users with the same amount of safety as the legitimate users. Unlike legitimate users, malicious users are always changing their accounts, making it almost impossible for the website or other users to track their activities [9]. But legitimate users who stay with one or just a few accounts can be traced because as those accounts keep being active within an online community, their reputation will grow and other users will be familiar with them, thus, only legitimate users are exposed to others. Effort has been made to address pseudonyms' low accountability. OpenID allows people to sign into other websites with an established account of OpenID issuing sites [13]. On the one hand, it saves the user plenty of time registering and managing account names and passwords. On the other hand, for OpenID relying websites, they do not need to put too much effort managing users' identities. In short, OpenID connects pseudonyms or we should say it reduces pseudonyms. Similarly, B. Ford and J. Strauss [14] propose that pseudonym parties could ensure everyone has only one pseudonym while maintaining the user's anonymity. But

more problems keep emerging because of pseudonyms. For instance, Yokoo et al. [15] study the false-name bids on auction websites. False-name bids are bids that are submitted by a single user but through different accounts. In their study, they show that there is no false-name proof auction protocol that is Pareto efficient, where every one gets optimal benefit.

2.2.2 Identity for OSNs

Identity for OSNs is paid great attention with the rising of social networks. Its unique property makes it distinct from the traditional online identity. The creation of OSNs such as Linkedin and Facebook, allows people to maintain an online identity within an overlapping online and real world context. These are often identities created to reflect a specific aspect or best possible version of themselves. Representations include pictures, communications with other "friends" and membership in network groups. Privacy controls, especially limited to specific networks like Facebook, are also part of social networking identity. Nowadays, many websites have integrated identity for OSNs such as Facebook, Linkedin and Twitter as their login method. Identity for OSNs consists of the user's profile in OSNs and the identity authentication to other web or mobile services. A user's profile can be regarded as the definition of the identity plus the user's relationship with others in the same OSN. What's inside profile? Name, gender, birthday, residence, interest, education, work, activity, family and friends etc., all of these personal data could be included in a user's profile.

Along with more and more people joining OSNs, identity for OSNs gradually has not been confined within the social networks. It has a couple of unparalleled advantages as an online social identity. To some extent, it combines the advantages of the online fixed identity and pseudonyms.

The OSN authentication, or broadly speaking, the online social identity is only based on the user's OSN account. There is no need for a user to provide her critical real world identity like social security number, driver license or passport to the website she wants to register. Furthermore, only the social network that is providing the social graph knows the user's whole information such as topology changes on the graph. The website to be authenticated would only obtain these information under the user's authority. Therefore, from the website's perspective, it is difficult for malicious users to forge identities, and

thus the online social identity protects the user's real identity better than the online fixed identity. On the other hand, compared to the pseudonym, since only OSN registered users can issue online social identities for themselves and users register with the social network in a secure way, one cannot ask the OSN to issue an identity for other users. It is true that a user can have multiple accounts in OSNs and it is the user's rights to choose an OSN account to authenticate. Thus, OSN identity is portable yet limited, unlike pseudonyms where each user has as many as different identities she wants over the Internet.

Like OpenID, the direct benefit for users of such social login plugins is to escape from the bothering to register new accounts. As most of them already have at least an OSN account, they only need to remember their social network accounts and log on other services with the same authentication information. Websites to be authenticated should welcome the social identity too, because at least they do not have to take an effort to build their own identity management system. OSNs or third party social identity providers take care of it[16]. The Internet will become a virtual community with the social identity. Users can decide whether to trust someone as it is in the real life, in contrast to pseudonyms with which new comers are likely not to be trusted.

An OSN not only contains the data that is needed to generate a social identity for a user, but it also knows a lot of information about a user. The real name policy that required by some OSNs increases the genuineness of users' information. Under the user's permission, the websites can make use of these priceless data such as email address, location and friend list for targeted marketing, advertising, customer survey and other business activities. Usually, the owner of a website, e.g. a furniture company or a supermarket, will also create an OSN fan page to cooperate these activities. M. Subramani and B. Rajagopalan [17] design a framework for virtual-marketing, where interested consumers can tell each other about the products, based on online social networks.

As pointed out in [18], the industry has adopted the concept of social networking and there is a growing need for a unified digital identity resource. For successful integration of social software into business processes, both reputation and authority have to exist. A trust model that can help in constructing community-aware identity management systems is proposed in [19]. According to Jennings and Finkelstein, incorporating social technologies within an organization has two key benefits: firstly, business processes can be improved through socially supported interactions; secondly, human knowledge can be captured and reused by the organization. With the accountability of the virtual community, it can expand

the scope of which we can rely on the Internet. For instance, we can order the food for taking out online. The user first logs into a restaurant's website using her OSN account and orders what she wants through the website. Once the store verifies that it is a valid resident in its community or service range through an OSN and probably plus the location based service, it can then prepare the food for delivery or for the user to pick up.

Making use of the OSN identity and the users' relationship on social network or other third party server, one user in an OSN can authenticate her friends in some emergency cases when they have left their tokens or forgotten their passwords. This "someone you know" authentication could be organized as a vouching process: first, the user asks a friend to vouch for her; second, the friend authenticates the user with some other ways and then issues some credentials for the requesting user; finally, the user uses the credential to log into the service. The vouching could be done either in physical way by contacting friends in the real world [20] or using mobile computing to automate the process [21].

2.3 Threats on Online Identity

Online identity plays a significant role in the internet community. Identity in OSNs is gradually becoming user's online "passport". Unfortunately, adversaries are also intended to exploit it for malicious purposes by launching various attacks, especially identity theft attack.

2.3.1 Typical Online Identity Theft Attacks

Identity theft attack is a term used to refer to fraud that involves stealing money or getting other benefits by pretending to be someone else [22]. It is a serious problem on the Internet today since it affects millions upon millions of people each year on both sides of the globe and it has only become worse as people communicate more frequently through the Internet. The eases with which we can send emails, make online purchases, and bank online have allowed our lives to be much more efficient, but they have also made us much more susceptible to identity theft attacks and their terrible effects. With so much of our personal information out there in cyberspace, it does not require much effort to gather enough information about a person and steal her identity. As a result, someone whose identity has been stolen may suffer various consequences when she takes the responsibility for the perpetrator's actions.

Identity theft attack is regarded by many as one of the major risks consumers and users are exposed to in today's digital environment. E-payment and online banking services, on which many people use every day, substantially suffer from such mistrust. You probably have heard that one of your friends lost her money due to the online identity theft or you could experience this yourself. In fact, in the United Kingdom, for example, an estimate of 3.4 million people had prepared to use the Internet but could not shop online because of a lack of trust or fears about personal security [23]. A survey targeting on the identity theft on Canadian online shopping industry has found that victims of identity fraud spent more than $150 million of their own money and spent 20 million hours to resolve the fraud in 2008 as part of a ballooning problem that struck almost 1.7 million Canadians [24]. Moreover, in China, a huge number of people have been suffering the QQ account theft attack. The author and his mother, both of whom are QQ users, have undergone this attack before. There is even an industry of hacking the QQ account in China, which provides adversaries the free software for hacking or helps them sell a good price of stolen accounts.

Although there are a number of ways to launch identity theft attacks, phishing is one of the most common approaches. The adversary attempts to acquire information such as usernames, passwords, and credit card details by masquerading as a trustworthy entity in an electronic communication. Communications purporting from popular chatting web service, auction sites, online payment processors or IT administrators are commonly used to lure the unsuspecting public. Phishing is typically carried out by e-mail spoofing or instant messaging, and it often directs users to enter details at a fake website whose appearance is almost identical to the legitimate one. Take the Bank of China's website for instance. In Jan. 2011, a large number of SMS-based web-phishing attacks targeted the Bank of China's online users. They received a phishing SMS that was designed to look like it was sent by the bank as a reminder to its customers: "Dear user, your token has expired, please visit http://www.boc**.com to reactivate your token." The URL is similar to the bank's official website but points to a phishing site that looks like the original bank website [25].

Another important way is malware. Short for malicious software, malware, consists of programming (code, scripts, active content, and other software) designed to disrupt or deny operation, gather information that leads to loss of privacy or exploitation, gain unauthorized access to system resources, and play other abusive behavior. Software is considered to be malware based on the perceived intent of the creator rather than any particular features. Malware includes computer viruses, worms, trojan horses, spyware, etc.

Marshall et al. [26] study online identity thefts. In their work, they point out methods of online identity theft: protocol weakness, naive users, malicious software, data acquisition and network impersonation. And pseudonyms make it just easier to steal someone's identity on the Internet. The identity theft attack is increasingly more sophisticated combining phishing, malware and other tactics to make credit card fraud and other abuses. At the same time a lot of technologies are developed against them. For example, tokens, certificates, dongles, etc., are devised specifically to protect against phishing. Many countries have noticed the problem and taken actions to help ensure that consumers and users are raising awareness about the identity theft attack and adequately protected against it [27]. These actions encompass consumer and user awareness campaigns, new or adapted legislative frameworks, private public partnerships, and industry-led initiatives aimed at putting in place technical prevention measures and responses to the threat.

2.3.2 Identity Theft Attacks for OSNs

Naturally, the identity theft attack techniques introduced above can apply to the OSN. However, because of its own property especially in privacy aspect, there exist some unique identity theft attacks in OSNs.

Privacy for OSNs

In order to address privacy concerns, OSNs allow users to hide their personal profiles from the public, e.g., Facebook and Linkedin's public and private profile. The same concern also goes to social identity authentication. When the user provides an OSN identity to a website, the website cannot gain the user's whole information out of the identity. The OSN needs to guarantee that the social graph is secured and not released to any third party including the government if there is no court permission. That means social identity is user-centric. As we briefly mentioned in the previous section, in any case, the user is the only person who asks for issuing her own OSN identity and decides to whom to release it, while websites can only verify an OSN identity through an OSN or third party provider.

Despite these protections, however, privacy issues can be widely seen in OSNs. In 2007, Facebook opened a development API, which allows developers to construct their own applications leveraging user profile data [28]. This was met with some concern for personal privacy. For example, one study reveals that applications written using this API could often

access significantly more information than necessary for their core functionality [29]. As an initial solution to this problem, Felt and Evans [29] propose a proxy-based architecture, which limits the amount of information available to installed applications. Singh et al. [30] propose a trusted third-party mediator called xBook.

Moreover, information security and privacy of OSNs have led to a number of broadly interesting research questions. In a research on human computer interaction (HCI), Wenday Mackay [31] has shown that only a minimal percentage of users tend to change the default privacy preferences which are highly permeable. In [32], they state that identity information released in a social network community, e.g. photo, politic views, course schedule, could also be used to identify people. He et al. [33] point in their work that personal attributes can be estimated especially for people who are closely related with other people. And now increasingly more people are aware of releasing personal information on the Internet and are more reluctant to interact with strangers. Krasnova et al. [34] define two types of threats that current OSN users are facing, organizational threats and social threats. As a response to organizational threats, users tend to disclose less information about themselves. Regarding to social threats, users tend to consciously control their released information.

Identity theft attacks for OSNs

The threats of OSNs are not confined to the potential risks however. Last year, identity theft was the main complaint received by consumer associations in the U.S., with about 9 million victims and related costs of more than $5 billion annually, according to the Federal Trade Commission of the United States [35]. Identity theft attack has plagued people for a long time. It is one of the fastest growing crimes in the United States and worldwide.

In the Internet world especially associated with the use of social networks, adversaries take advantage of these data exposed in the OSN identity to launch various identity attacks. OSNs may also enable new forms of classical attacks, including phishing [36] and spam [37]. For example, a phisher can easily and effectively exploit the information available on social networks to increase the success rate of a phishing attack. The email phishing attacks can be achieved 72% hit rate by using the information available in the social network [36]. OSNs are also vulnerable to social engineering techniques which exploit low entry thresholds to trust networks, and to scripting attacks that allow the automated injection of phishing links. It is threatening to users by revealing the sensitive information. Social relations in

OSNs can also be faked, which causes a new problem called social graph privacy [38]. J. Bonneau et al. [38] investigate how malicious users would leverage the valuable information by reconstructing social graphs to issue attacks. Even if people choose not to disclose their private information, it can still be inferred. Furthermore, these identity theft attacks can be automated. Bilge et al. [6] present how easy automated crawling and identity theft on popular social networks. They propose two ways of profile cloning, direct profile cloning and cross-site cloning as we mentioned in Section 1.1.

The OSN online identity can be seriously destroyed and taken advantage of by adversaries through various ways such as financial crime, reputation destruction or behavior tracing. Additionally, the whole application system built on the social identity would be shaken. For example, in Latin America, social networks like Facebook, Twitter and Myspace, are used by identity thieves, who often create fake profiles and connect to thousands of people, accessing their personal information. In the case of Facebook, fake profiles are very common and can be easily mistaken for real users who end up sharing all types of personal information with strangers. A recent survey by Sophos [39], a multinational security company, shows that 41% of Facebook users disclose personal information such as email address, birth date and phone number to complete strangers, thus increasing the likelihood of identity theft. According to the study, 46% of Facebook users have accepted friend requests from strangers. In effect, accessing personal information about people on Facebook is so easy that Freddi Staur, a plastic green frog, was able to accomplish that [39]. A separate study of Facebook users done by Gross and Acquisty reveals that 71% of the Facebook users have the tendency to provide large amounts of sensitive personal information in their profile that expose themselves to various kinds of security risks [40]. Furthermore, with the advancement of the data mining technology and the reduction of cost of disk storage, the third party can create a digital dossier of personal data with the information revealed on the profiles of OSNs. A common vulnerability is that more private attributes which are directly accessible by profile browsing can be accessed via search.

Another phenomenon is an adversary is intended to create a fake profile to impersonate a renowned person or a brand. This is particular common in microblogging OSNs like Twitter and Sina Weibo (a Twitter like service in China) which focus on fast content sharing among the users and are lack of strict real name policy. Take Sina Weibo for example. It had 250 million users as of September 2011 [41] and due to its property of social media, the public celebrities dominate the message spreading. This phenomena attracts the malicious user

to register the renowned person's name before the real person or simply forge the account of a famous guy or organization. Typically, the malicious user would post some "real" stuff to appeal more "followers" which can also be faked. As soon as there is a large number of the followers, he can post some irresponsible messages, rumors and advertisement or just sell these accounts to some entities that want to promote themselves.

2.4 Related Work

With respect to defending against identity theft attacks in OSNs, some solutions focus on educating users to control the distribution of their sensitive personal information and digital identities e.g. FightIDTheft [42], a website that concentrates on providing detailed suggestions to help users define their privacy policies. There are several third-party applications of OSNs proposed and employed for protecting users against identity theft attacks. For instance, in Facebook, identity badge [43] identifies a user via a passport check, and mysafeFriend validates a user's identity by asking the user's friends to verify her and doing a credit card check [44]. These applications may help users validate who their "friends" are and protect their identities. However, many users would not bother to show their real IDs which might cause further privacy issues. Besides it is a passive protection meaning that the forged identities still exist in OSNs and adversaries continue to deceive more victims using them without any restrictions.

In [7], an active approach to detect the identity theft attack is proposed. They considered two important aspects, several people having similar names in the real world and characteristics of a fake identity. Although they have demonstrated through simulation that their detection schemes are feasible and effective, it is not easy to estimate the parameters in their models to make sure they can work well when putting them to far more complicated real OSNs. L. Bilge et al. suggested that it can be helpful to improve the security of contact requests by providing more information to the receiver on the authenticity of a request and the user who is sending it. For example, the social site could send extra information on where the request was issued such as IP address and the profile creation date. Unfortunately, we have not seen an implementation for that idea so far. Therefore, to fight against the identity theft attack, it is necessary to build something effective and easy to deploy without changing the current infrastructures.

When it comes to OSN providers, they do worry about this issue. Typically, well-

designed OSN sites allow users to customize their privacy policies. For example, Facebook has a "Privacy Settings" page that allows users to specify which pieces of profile data everyone can see. It also allows users to set fine-grained access policies by specifying whether a piece of profile data is visible or not to a certain friend. However, configuring fine-grained privacy settings in OSNs are often complicated and time consuming tasks that many users feel confused about and usually skip. As a result, most of users are prone to stand on the side of popularity and usability rather than security and privacy. Furthermore, Facebook has a policy that urges users to provide truthful information when registering. "Truthful information" includes no fake name, accurate contact information and no false personal information. And under its "Registration and Account Security" section, there are rules that users will not create account for other people, will not create more than one account and will not transfer their accounts. Moreover, Facebook reserves the right to shut down an account if any of the rules were breached. And it did cancel users' accounts for above reasons. With its real name policy, Facebook is trying to maintain a more accountable Internet environment. However, this policy does not always go well because to detect a fake account is not easy. An interesting example is about a famous writer Salman Rushdie [45]. When he wanted to change his name on his Facebook profile, the social site deactivated his account over a couple of days saying "We don't believe it was you". For Sina Weibo, they also have a set of manual examination procedures when somebody claims that she is a renowned person. To be verified by the Weibo administrator, one needs to proof that it is her or him by uploading photocopy of identities like passport. But this examination is not mandatory but voluntary. In addition, unlike the identity theft in the real world, it is highly difficult to catch the adversaries in the Internet community in terms of that it is a visual world.

Unfortunately since the existing approaches can not perfectly handle the identity theft attack, for an ordinary user, one way to protect your identity in social networks is to check privacy settings and restrict access to published content including uploaded photos. Another advice that seems obvious is to only accept friends requests from people you know. But the problem is we cannot push all the responsibility to the OSN users.

In the end, we might ask ourselves: How do we know if the person requesting my friendship on Facebook is the one she or he claims to be? For this problem we launch a mobile device based identity validation system called CredFinder for OSNs that can greatly help the user verify friends as well as herself.

Chapter 3

System Design

This chapter states the design objectives of our identity validation system, the architecture of the proposed scheme, the detailed protocols and the advantage of our system. I name our identity validation system "CredFinder" which means our system helps users find out their real friends in OSNs via users' credentials .

3.1 Design Objectives

Many people have such an experience in an OSN (e.g. Facebook) that when you receive an "add friend" request from somebody, it is difficult for you to decide whether or not to accept that request because you are not sure if this is from a fake identity. Moreover, sometimes, you even have no idea who she or he is. Based on people's actions in the real world, we categorize into three scenarios in CredFinder when A receives the adding friend request in an OSN from B.

- *Scenario 1*: Receiver A and requestor B have met "recently".
- *Scenario 2*: Receiver A and requestor B have not met "recently" but one or more validated friends can link them together.
- *Scenario 3*: Receiver A and requestor B have not met "recently".

To the designer's experience, these three scenarios cover most cases when validation happens. For example, when a professor gets a friend request on Facebook from someone who claims to be a student in his course, it is highly possible that this professor does not

know who he is because there are so many students in the class. However, they have met recently since the professor is offering the lectures every week. So this case can fall into Scenario 1. For other scenarios, the typical instance would be that two long-lost primary school friends Alice and Bob happen to contact each other in an OSN and they live in two different cities or even countries. If they can be linked together through other validated friends in CredFinder, it falls into Scenario 2. Otherwise, it falls into Scenario 3. We will explain each scenario in the following protocol section. In short, our identity validation system is able to benefit the user who faces the scenarios above by validating the OSN requests.

3.2 Architecture of CredFinder

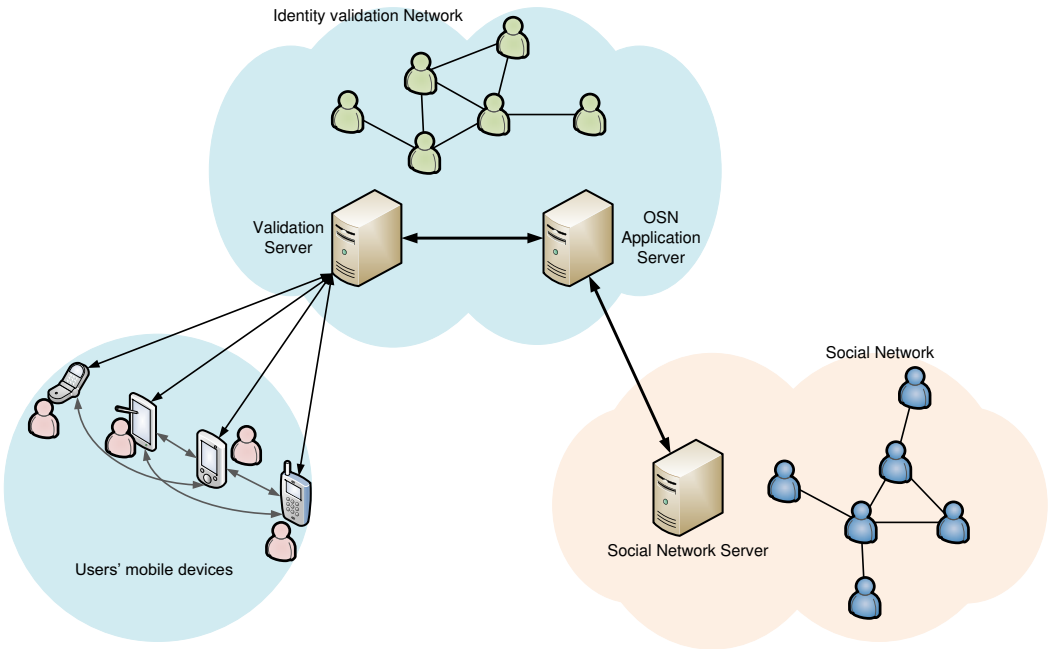

Fig. 3.1 System architecture

The natural way to enhance the security of contact requests is providing more informa-

tion to the receiver of the request as well as the user who is sending it [6]. We believe the most valuable and truthful information is one's relation in real life such as when and where she meets with others. Our design principle is to allow our users to control their credentials (also referred in this thesis as "identity tokens" or simply "tokens") and identities, and to enable simple, secure and trustworthy access to our services. Since mobile devices (e.g. smart phone, tablet computer) represented by Android and iPhone are becoming more and more pervasive, they can be used by people to carry identity tokens. The architecture and major components of the CredFinder are shown as Figure 3.1.

Mobile devices: We regard the mobile device as an "identity aware device" which represents its owner. The role of the mobile device is to: (1) help users to handle their identity tokens; (2) mediate and simplify users' interactions with validation server. Here are the main tasks the mobile devices need to process.

- Acquiring the identity tokens passed by the validation server.
- Transferring identity tokens to other devices.
- Uploading the identity tokens received from other devices to the validation server.

Validation Server: The validation server is the core processing node of CredFinder. The functions of the validation server are as follows.

- Distributing/receiving tokens to/from users' mobile devices.
- Formulating the identity validation network by maintaining users' token exchange lists (friend lists).
- Providing the identity validation network graph to the OSN application server.

OSN Application Server: The application server sets up an OSN application website to let the users apply the identity validation network to help them verify the people they are going to connect in the OSN. The roles of the OSN application server are:

- Offering an interactive interface of the identity validation network to users.
- Transmitting and displaying tokens upon user's request.

3.3 Protocols

This section states the protocols for Scenario 1, 2 and 3 respectively. In protocol 1, I give the full description, whereas in protocol 2 and 3 only the different parts from protocol 1 are pointed out for the purpose of brevity.

3.3.1 System Requirements and Assumptions

As a prerequisite for our identity validation system, the user has to possess a certain OSN account and a smart mobile device with the following features.

- Bluetooth support.
- Internet access.
- A-GPS support (optional but preferred).

3.3.2 Protocol 1

Fig. 3.2 Users' devices interaction of Protocol 1

1. App Installation: User Alice installs the CredFinder application on her mobile device such as a smart phone A and allows the location service on the device.
2. DH key exchange: Device A contacts the validation server, and initiates a Diffie -

Hellman (DH) key exchange via an Internet protocol. The rest of the communication between Device A and the validation server are encrypted and decrypted by the shared key.

3. User login: Alice logs in herself to our identity validation system.

4. Token generation: The validation server randomly generates a number S and hashes it using a hash function like MD5 or SHA1/2 to obtain $H_1(S)$. Then it takes the hash of the hash to obtain $H_2(S)$ and repeats this n times, to obtain $H_3(S)$ through $H_n(S)$, where we say n is 100. Afterwards, Alice's unique id assigned by the validation server is attached to each hashed number and these two parts will be encrypted by the server's public key. We define each encrypted string as an identity token. These tokens are stored in the validation server and will be re-generated and re-distributed when she runs out of them.

5. Token distribution: The validation server transmits those tokens belonging to Alice to Device A.

6. Meet the token exchange candidates: When Alice is in a social activity such as a business meeting, a causal chatting, a birthday party or even a class, every other user who has not exchanged tokens with Alice before and is present in the proximity of Alice, according to the location service, will appear on Alice's token exchange candidate list.

7. Alice sends token and records status: If Alice wants to send a token to any candidate e.g. Bob, Bob needs to turn his mobile device in Bluetooth 'discoverable' mode upfront. Alice then opens the application and clicks Bob's icon on her phone and her phone will connect Bob's via Bluetooth and send a token along with her name, the current time and location. Meanwhile it stores Bob's name in the phone labeled "token sent" and removes Bob from her candidate list.

8. Bob replies Alice with his token: On receiving Alice's token, a request will be shown on Bob's screen asking him to reply with his token or not. If Bob says yes, his device would send back a token with his name, the current time and location to Alice. His device at the same time stores Alice's name locally labeled token sent and removes Alice from the candidate list. If Bob says no, his device would do nothing.

9. Bob and Alice upload the token to the validation server: Once a user receives the token, besides making a possible response to the sender, she or he also uploads the token to the validation server.

10. Validation server updates Bob's received token list: When the validation server receives token from the user, it will store it in the list under the name of that user.

11. Bluetooth shuts down: After a certain time period (in our case 5 minutes) from the user sending the last token, her or his device's Bluetooth will be automatically shutting down for saving the power.
12. Alice adds Bob as a friend in an OSN: Some times later, when Alice wants to add Bob as a friend in an OSN, she could send Bob both the regular request and the additional message like "we met at McConnell Engineering Building at 2:05pm on Dec. 2th, 2011" according to her token from Bob on the validation server.
13. Bob responses: Bob would compare the token shown by Alice with his own sent token list on the validation server to check out whether this token was truly sent by himself. No matter what Bob finds out, true or false, it will help him make a decision on whether or not to accept the request. Note that this procedure still works with one side token (i.e. Alice sends token to Bob but does not get response).

Several notions and design considerations should be explained here. According to this protocol, the word "recently" in design objectives means people have met and exchanged tokens before. The token involves two parts, the user's id to indicate who owns this token, and the hashed number to ensure each token is unique. We do not use the user name to distinguish one another as multiple persons can have a same name. When the user uploads his or her received token, the token exchange time and location are also uploaded simultaneously. This information aims to help the user recall the token exchange scene later on. In addition, tokens could be generated on the user's mobile device, however, considering its computing power and battery life are limited, we put the token generating part on the validation server. Also, there are a couple of ways to generate the hashed numbers, our system just applies one of them. Moreover, a public-key cryptography algorithm is definitely an alternative to symmetric-key algorithm used in the device and server communication. But to avoid the complexity of public-key computations, we have chosen the symmetric-key algorithm.

In step 6, we mentioned that we use the location service in our system. CredFinder exploits it in two aspects. One is that the location service reports to the user that who's available for the token exchange around the user with a visual impression. So the user does not need to ask the person who she meets that "Are you also in CredFinder?" or "Would you like to exchange the token with me?". The other is that the location service maximizes the power efficiency of the user's mobile device. Once the location service detects that there is no one available around, it will notify the user and help her shut down the Bluetooth to

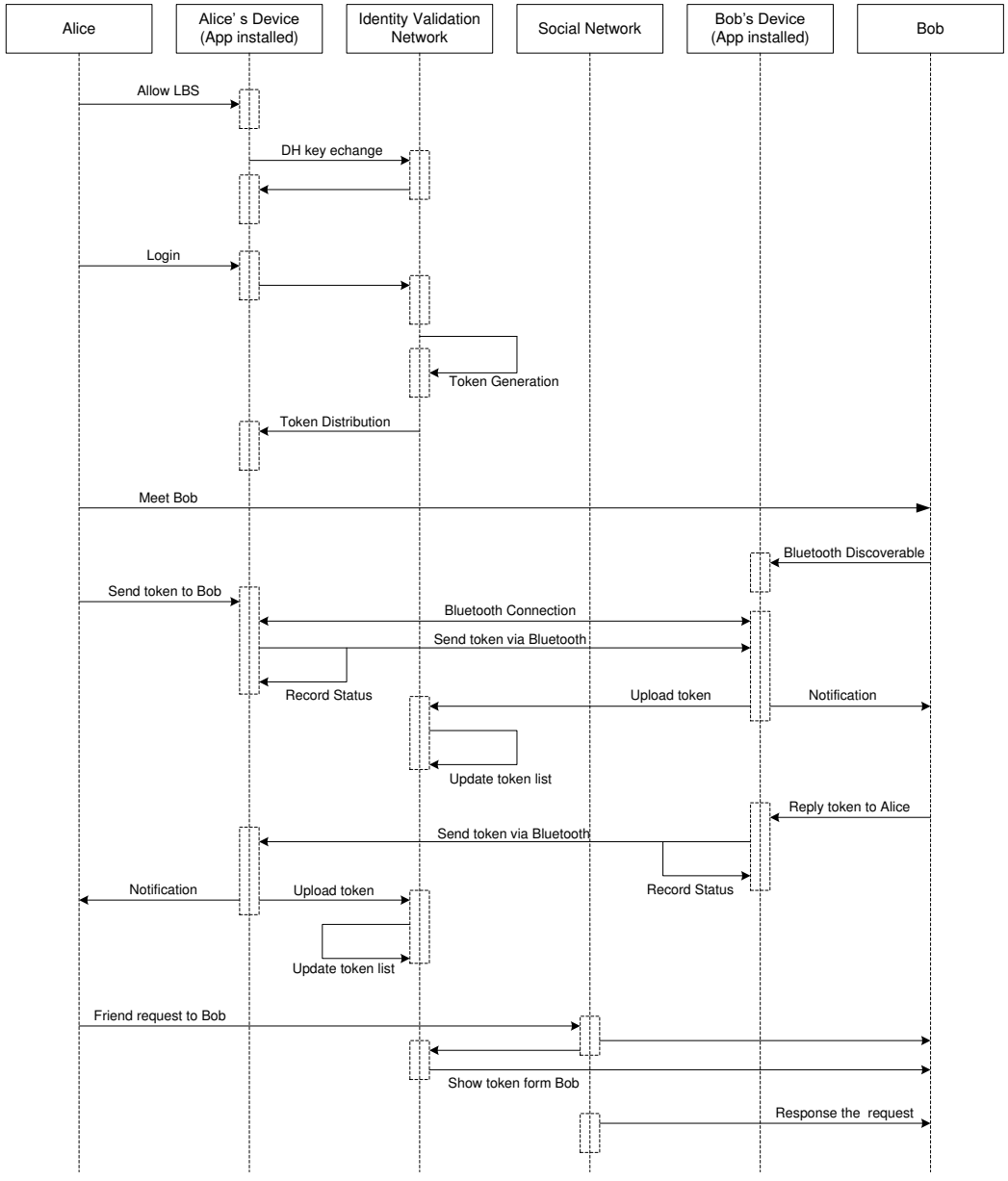

Fig. 3.3 Graphical representation of Protocol 1

save the power of the device. Furthermore, the location service is planed to automate the token exchange process in next version of the system.

A graphical representation of protocol 1 is shown in Figure 3.3, along with the sequence of events and messages between the different entities involved in the CredFinder system. It is worth mentioning that this protocol is from the perspective of the user Alice. The configuration and the possible request from Bob are ignored.

3.3.3 Protocol 2

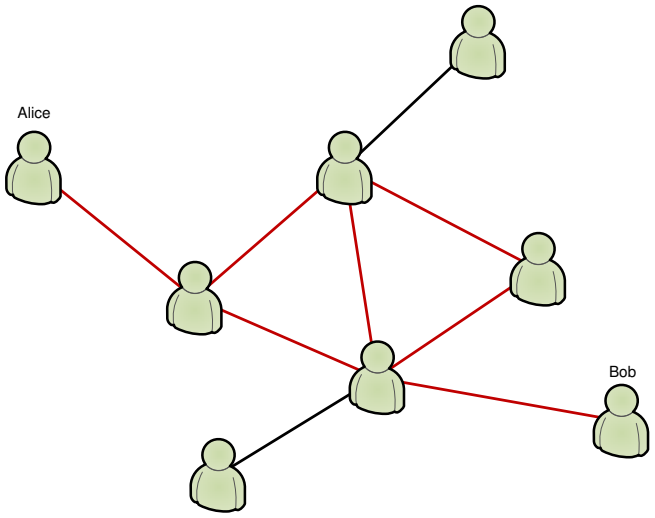

Fig. 3.4 The connection of Alice and Bob in CredFinder of Protocol 2

In this scenario, two persons, in our long-lost friends example Alice and Bob, did not meet physically, but by checking the identity validation network Alice knows that she and Bob are linked together in our validation network via other users (See Figure 3.4). As long as they can be connected in our validation network, the hops between them could be from two up to a large number as the red lines show in the graph.

Step 1 to step 11 of this protocol are the same as protocol 1. The remaining steps are as follows.

12. Alice adds Bob as a friend in an OSN: Alice sends Bob the regular request and

the indirect link information in the identity validation network (An analogy of this is in Linkedin people are provided the social distance between one another).
13. Bob responses: When he receives the friend request and connection information from Alice, he would check it out and make his decision on whether or not to accept the request.

Predictably, the link distance and the number of people Bob knows on this link would affect his final decision. The shorter the distance and the more people Bob knows, the more likely that he will accept the request. Also, with the growth of this distance, more security risks could appear.

3.3.4 Protocol 3

In this scenario, neither did Alice physically meet with Bob nor there is an indirect link between them in CredFinder. However, Alice indeed has exchanged tokens with some of the people Bob may know in real life though he has not exchanged tokens with those people. They could be, for example, Alice and Bob's common friends but live in the same city with Alice. This case is illustrated in Figure 3.5. Again, step 1 to step 11 of this protocol are the same as protocol 1. The remaining steps are as follows.
12. Alice adds Bob as a friend in an OSN: This time, when Alice wants to add Bob as a friend in an OSN, she could send Bob the regular request plus the people's names who have granted their tokens to her provided by the validation server. Alice takes charge of who and how many tokens are presented to Bob.
13. Bob responses: When he receives the friend request and those additional information from Alice, he would check it out and make his decision on whether or not to accept the request. Obviously, the more tokens Alice got from the persons Bob knows, the more likely Bob accepts her friend request. The tricky problem here is Alice has to speculate who Bob may know, and thus sends him the corresponding tokens. Take another concrete example. Alice, a student of McGill, wants to add Bob, another student in McGill but from a different department. If Alice could know Bob's department and which year he is in, she could send the tokens of the guys from his department or classes. But if Alice cannot gather enough information to speculate, it is a little bit difficult to use this protocol.

It can be seen that, the "distance" between Alice and Bob is two because there is one hop between Alice and their common friends on CredFinder and one hop between Bob and their common friends in the real world. In some cases, both protocol 2 and 3 can be

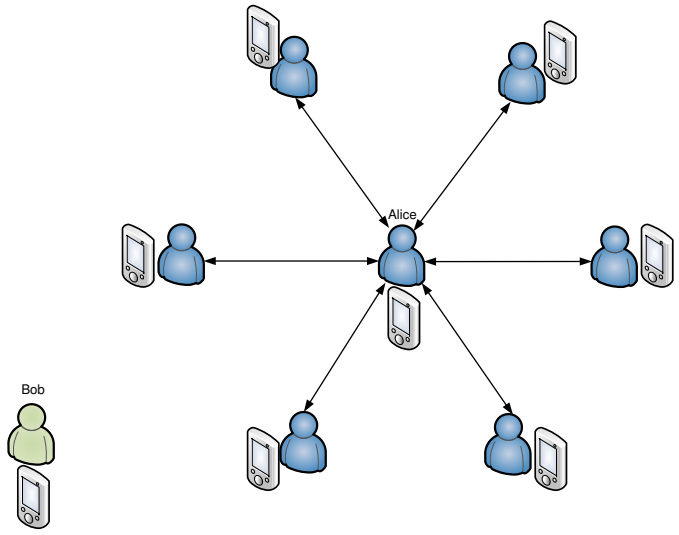

Fig. 3.5 Users' devices interaction of Protocol 3

applied, separate or combined. Additionally, this protocol can also help users especially the celebrities (See Sina Weibo example in Section 2.3.2) to claim themselves among the forged IDs. What the user needs to do is to show his or her tokens collected from others when registering a profile in a certain OSN.

3.4 System Advantages

If Alice and Bob have exchanged the token before, when an adversary Andy claims to be Alice in an OSN and tries to add Bob as a friend, Bob has enough reason to suspect that this "Alice" is a forged one and takes further actions to confirm his judgment and to report the adversary. It can be seen that our identity validation system prevents both the victim and her friends from the identity theft attacks. Even if Alice and Bob have not exchanged the token before, Bob could ask Andy to show the token exchange status between Andy and persons he knows, or check out our system to see whether they can be linked together in the validation network. If Andy could not provide data in either way above, again it is likely that this is a fake profile. In summary, as a practical system, one of the biggest

advantages of CredFinder is that it can comprehensively protect the user from the identity theft attacks.

Considering the user, the mobile device, the identity validation network and the interaction of them, other advantages of our system list as follows.

- *Effectiveness*: CredFinder as an identity validation system is formulated based on the people's real world connections. It can effectively prevent the user from the attacking of adversaries because credentials in the real world are trustable and hard to forge for the adversaries.
- *Accessibility*: By installing our mobile application and clicking a button, the user can build its own real life friend network for further validation in OSNs. Also, the protocols we devised are trying to minimize bothering the user by doing most of the work in the back-end.
- *Privacy Protection*: Although this system makes use of the real world connection to do the validation, the user's real world information is not needed. Our system, which is privacy preserving since it does not need revelation of a user's cell phone number, is based only on proximity sensitive connections between users' mobile devices as well as the connections from mobile devices to the validation server.
- *Cross-platform*: CredFinder does not have to associate with a particular OSN. It can apply to all the OSNs so that it protects the user from cross-site identity attacking. With respect to the mobile device, a specific user is able to use multiple devices belonging to her as long as she uses the same user ID.
- *Adaption*: Unlike the theoretical approach, CredFinder can be easily applied to any OSN without doing parameter estimation or other complicated calculations.

3.5 Using Mobile Device As a Token

The core concept driving our system is using a mobile device as a token, or precisely speaking it stores the user's identity tokens. Making use of users' mobile devices, we rebuild the real life social networks in the virtual world just like what the business card does but in a more portable and powerful way.

3.5.1 Bluetooth and Its Alternatives

Bluetooth is a proprietary open wireless technology standard for exchanging data over short distances (using short wavelength radio transmissions in the ISM band from 2400-2480 MHz) from fixed and mobile devices, creating personal area networks (PANs) with high levels of security. Bluetooth today is a standard peripheral in most mobile devices. To accomplish the token exchange task, it is our primary choice. People who are intended to exchange the tokens are usually in a meeting, party, office where they are close enough so that Bluetooth, whose transceiver range is typically around 10 meters, is a good option because it can quickly and efficiently send and receive the tokens.

A good argument here would be that other transmission methods such as short message service (SMS) are also feasible for the token exchange. For SMS, the problem is that SMS traffic may be snooped. Although encrypting the SMS probably can solve it, it is more expensive and in most cases unnecessary compared to Bluetooth. Bluetooth's effective range (typically 10 meters) implies that it is very hard for the malicious users to attack. However, it is possible that we can use SMS as an assistant token exchange method.

Another alternative device communication approach is near field communication (NFC). NFC creates a new and universal interface among existing devices through simple touch interaction within a very small area, anywhere from a touch to four centimeters [46]. NFC is able to replace the pairing of Bluetooth-enabled devices, or the configuration of a WiFi network through PINs and keys because of its substantial simplicity of use while the level of confidence is similar. NFC enabled phones can be used as contact-less smart cards, and thus offer means of token transmission using mobile devices. However, although NFC phones are quickly increasing in high-end market, they are far away from widely used. What we need is a solution using commodity devices currently widely available.

3.5.2 Using Location Service

The location service greatly assists our system. A location service is an ability to make use of the geographical position of the mobile device, accessible with mobile devices through the mobile network or A-GPS. It includes services to identify a location of a person or object, such as discovering the nearest banking cash machine or the whereabouts of a friend or employee. Nowadays, it is built almost in all of the smart phones and tablets. In some cases like location based authentication which involves determining the user's location

and making access conditional to it, it is only useful when the location is controlled and physically secure. This would apply to restricted military installations and server farms where it is necessary to determine the user's proximity in the restricted area. This can be done with a trusted hardware sensor placed at the site. Sources of location information can be GPS, cell towers, beacons and proximity sensors. GPS based location is mainly available in open areas as it requires line of sight access to the satellite signals, although some sensitive GPS devices have some reception indoors, especially near windows. Cell tower information gives coarser location information but generally works indoors. While it does not have global coverage, thus excluding very rural or remote users, it usually covers places where most of the population lives. Beacons and proximity sensors are not applicable to our case, so we save the words to introduce them.

Chapter 4

Implementation

In this chapter, I give some of the pertinent details of our prototypical implementation of CredFinder. According to the system architecture, the implementation is inherently divided into three main entities—the mobile device application, the validation server and the OSN application server. It should be noted that although we described three slightly different protocols handling various scenarios in the design phase, in the implementation, these three protocols are seamlessly integrated together from the user's perspective.

4.1 Overview

Our ultimate goal is to deliver our mobile application in every major mobile operating system so that whatever smart phone the user owns, she can access and enjoy our system. However, as the debut of our system, we choose Android, one of the fastest growing and most popular mobile platforms powering millions of phones, tablets, and other devices. Besides, it is an open source software. Not only need we to select a mobile platform, but also an OSN service to "help". Without a doubt, we target on Facebook, the dominant social network in the world having more than 800 million active users.

The validation server in the current prototype is a Java program. For the OSN application server, it is powered by Drupal, an open source content management platform built on LAMP (Linux, Apache2, MySQL, PHP) web service architecture.

4.2 Mobile Device Application

The mobile application takes the responsibility of interacting with users, running the token transmitting protocol and communicating to the validation server. It should be installed by the user from the Android application market if the application is published.

4.2.1 Android Platform

Propped up by Google Android libraries and application framework, Android applications are written using Java. Once installed on a device, each Android application lives in its own security sandbox. There are in general four crucial components in order to build an Android application [47].

- *Activity*: An activity represents a single screen with a user interface (UI).
- *Service*: A service is a component that runs in the background to perform long-running operations or to perform work for remote processes.
- *Content provider*: A content provider manages a shared set of application data.
- *Broadcast receiver*: A broadcast receiver is a component that responds to system-wide broadcast announcements.

Furthermore, there are some distinct features in Android programming. One of them is called "Intents" which are messages activating activities, services, and broadcast receivers. Intent messaging is a facility for late run-time binding between components in the same or different applications. The Intent itself, an Intent object, is a passive data structure holding an abstract description of an operation to be performed [47]. Additionally, the Andoid UI toolkit is not thread-safe. UI cannot be manipulated from a worker thread meaning all manipulation to the user interface must come from the UI thread. Thus, there are simply two rules to Android's thread model: 1) Do not block the UI thread; 2) Do not access the Android UI toolkit from outside the UI thread. What's more, when a process is created for the application, its main thread is dedicated to running a message queue that takes care of managing the top-level application objects. Therefore, a child thread should send message to the father thread's "Handler" to complete the threads communication. A "Handler" class allows you to send and process "Message" and "Runnable" objects associated with a thread's "MessageQueue". Each Handler instance is associated with a single thread and its

message queue. When you create a new Handler, it is bound to the thread message queue of the thread that is creating it. Considering these and other properties of the Android platform, we have built our CredFinder mobile application.

4.2.2 Structure of Mobile Application Program

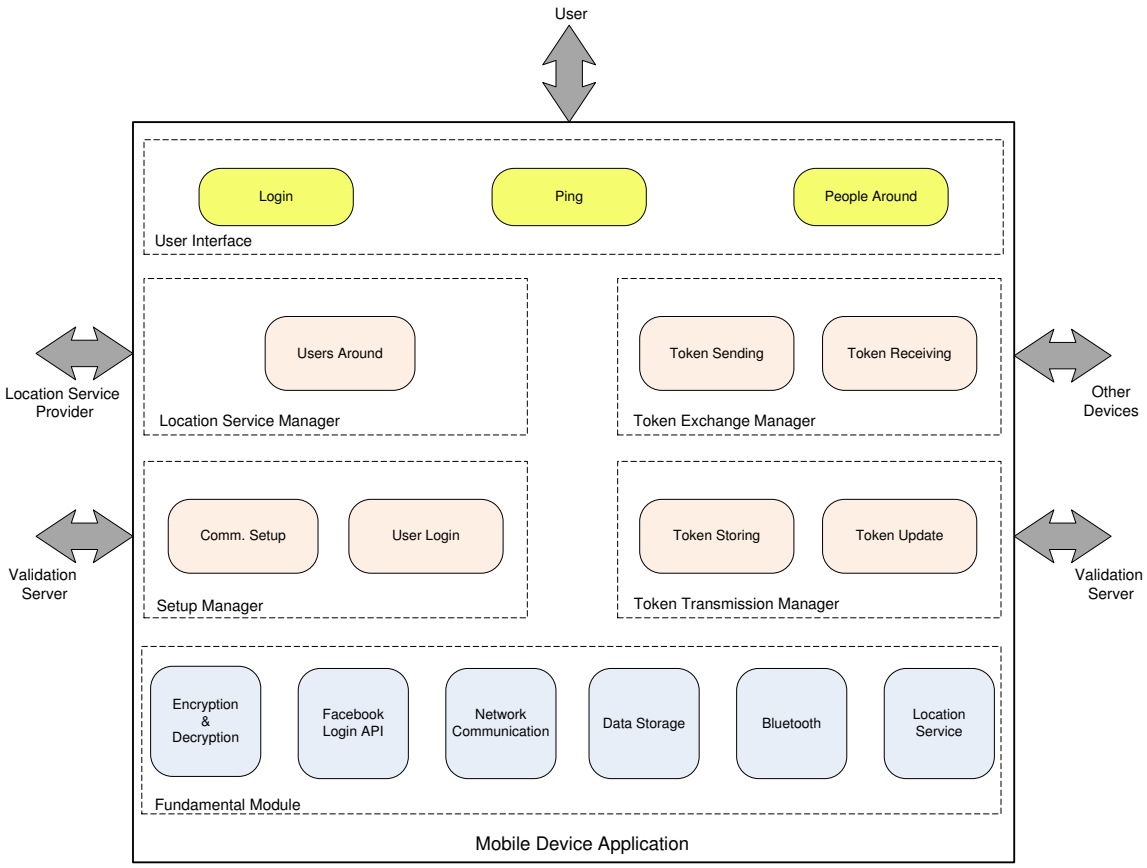

Fig. 4.1 Components of the mobile device application

Figure 4.1 illustrates the structure of the mobile device application program. The following subsections describe the deployment and the adaptation of these building blocks from bottom to top in details. This general description of three layers identifies the following key roles and entities:

- *Fundamental modules*: The basic building blocks of the program.

- *Local managers*: Four managers to handle the user tasks using the fundamental modules.
- *User interface*: Providing the interface for the user to manipulate the program.

These building blocks are java class files. In this context, for the purpose of revealing the inner relationship between each part, we present our program in three layers.

4.2.3 Fundamental Modules

Network Communication

Network Communication module is the basic module supporting the communication between the mobile device application which is the client side and the validation server that is the server side. Implemented by Java Socket API using TCP/IP stack, it guarantees a reliable data stream service being used by the setup manager, the token transmission manager and probably the location service manager.

Encryption and Decryption

Encryption and decryption component ensures that the data exchange between the mobile device and the validation server is achieved in a secret protected channel. Diffie - Hellman (DH) key exchange provides the foundation of the encryption for the rest of the communication. It is one of the earliest practical examples of key exchange implemented within the field of cryptography. The DH key exchange method allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher. The algorithm is illustrated in [48]. The protocol has two system parameters p and g. They are both public and may be used by all the users in a system. Parameter p is a prime number and parameter g (usually called a generator) is an integer less than p, with the following property: for every number n between 1 and p-1 inclusive, there is a power k of g such that $n = g^k \bmod p$. Suppose Alice and Bob want to agree on a shared secret key using the DH key agreement protocol. They proceed as follows: First, Alice generates a random private value a and Bob generates a random private value b. Both a and b are drawn from the set of integers. Then they derive their public values using parameters p and g and their private values. Alice's public value is $g^a \bmod p$ and

Bob's public value is g^b mod p. They then exchange their public values. Finally, Alice computes $g^{ab} = (g^b)^a$ mod p, and Bob computes $g^{ba} = (g^a)^b$ mod p. Since $g^{ab} = g^{ba} = k$, Alice and Bob now have a shared secret key k.

We need to set up this secure channel before we transmit the data. First, shared keys are generated and exchanged according to the DH key exchange algorithm. And then we employ AES-128 for the encryption of the following data. It is worth noting that it is necessary to encode the data using Base64, a group of similar encoding schemes that represent binary data in an ASCII string format by translating it into a radix-64 representation, before sending data to the communication channel. Similarly, the inverse part applies to receiving the data. Moreover, to secure the tokens' storage and exchange between the mobile devices, another layer of encryption is needed. Therefore, we apply validation server's RSA public key to every token before distributing it to the user's device. In practice, we use a set of APIs from Java security and cryto packages to accomplish the algorithm desired in this module.

Facebook Login

To access CredFinder, our users need to log in the system first. We could build our own identity management system to authenticate the users. However, we found out that using the Facebook authenticated referrals could benefit both the CredFinder developer and the user. Because on the one hand, we do not have to make an effort to build an identity management system. In addition, since our prototype targets Facebook, we can take advantage of the users' data in Facebook; on the other hand, users are not bothered to register a new account. As we believe in simple and elegant design, we deploy the authenticated referrals. It ensures all referral traffic from Facebook to the application is already connected with Facebook. This means that the visitors arrive on the application are already registered with whatever data permissions (email, likes and interests, etc.) we request in the required permissions section. We can use this information to provide a personalized experience for Facebook visitors when they are using our application.

The authentication method needs to use Facebook Android SDK defined in Facebook.java in which we implement these methods to do the following [28]:

- Initialize the Facebook instance with our application ID. Once initialized, the instance can be used for follow-on method calls.

- Initiate the user authentication and application authorization flow. Once this is completed successfully we can make additional API calls that require the access token.
- Handle the Single Sign-On URL (SSO) callback. This is part of the SSO implementation.
- Log out the user from Facebook.
- Check if the user session is valid. To do this, we can check if an access token is valid so we can handle re-authenticating the user, or we can expose functionality that requires the user to be logged in.

Since the application can keep user's login status stay, typically the user only needs to log herself in at the first time she uses the application.

Data Storage

Android provides several options to save persistent application data. The chosen solution depends on the specific needs. Except content provider we mentioned in Section 4.2.1, the data storage options are the following [47]:

- *Shared Preferences*: Store private primitive data in key-value pairs.
- *Internal Storage*: Store private data on the device memory.
- *External Storage*: Store public data on the shared external storage.
- *SQLite Databases*: Store structured data in a private database.
- *Network Connection*: Store data on the web with given network server.

In our case, we need to store the tokens received from the validation server as private data confined within our application so that neither other applications nor users can access them. We save these tokens in a file on the device's internal storage and they cannot be removed unless the user uninstalls the application. These tokens will be sent by the user if she meets the persons she wants to exchange them with.

Bluetooth

The Android platform includes support for the Bluetooth network stack, which allows a device to wirelessly exchange data with other Bluetooth devices. The application framework provides access to the Bluetooth functionality through the Android Bluetooth APIs enabling point-to-point and multipoint wireless features.

The Bluetooth can make an Android application perform the following [47]:

- Scan for other Bluetooth devices
- Query the local Bluetooth adapter for paired Bluetooth devices
- Establish radio frequency communication (RFCOMM) channels
- Connect to other devices through service discovery
- Transfer data to and from other devices
- Manage multiple connections

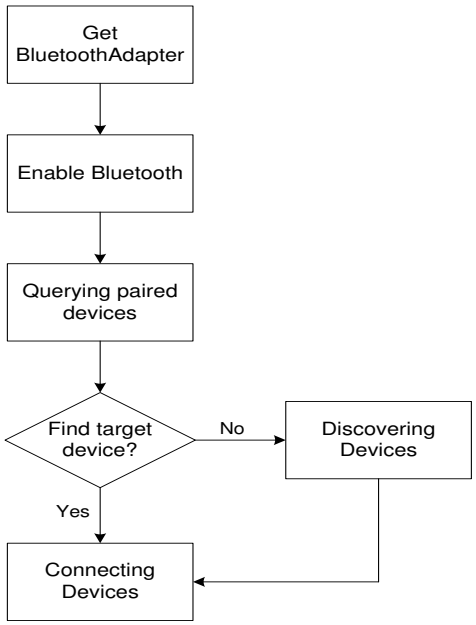

Fig. 4.2 Bluetooth connection procedure

Figure 4.2 illustrates the connection procedure before exchanging the data. First, we need to verify that Bluetooth is supported on the device, and if so, ensure that it is enabled. Then, using the Bluetooth adapter, we can find remote Bluetooth devices either through device discovery or by querying the list of paired (bonded) devices.

Before performing device discovery, it is worth querying the set of paired devices to see if the desired device is already known. If it is, we could directly go to the connecting stage;

otherwise, device discovery is necessary. Device discovery is a scanning procedure lasting 12 seconds that searches the local area for Bluetooth enabled devices and then requesting some information about each one. In order to make a Bluetooth device within the local area respond to a discovery request, we need to enable the discoverable upfront. In our application, to balance the power efficiency and user experience, the device will become discoverable for 120 seconds. Thus, it will respond to the discovery request by sharing some information, such as the device name and its unique MAC address. Using this information, the device performing discovery can then choose to initiate a connection to the discovered device. Once a connection is made with a remote device for the first time, a pairing request is automatically presented to the user. When a device is paired, the basic information about that device is saved and can be read using the Bluetooth APIs. Using the known MAC address for a remote device, a connection can be initiated with it at any time without performing discovery (assuming the device is within the connection range). After the above procedures, to create a data connection between our application over two devices we must implement both the server-side and client-side mechanisms similarly as the TCP/IP socket communication. In the end, the token exchange can be done upon this RFCOMM channel.

Location Service

Our application needs to access the location services supported by the device through the classes in the android.location package. The central component of the location framework is the location manager system service, which provides APIs to determine location and bearing of the underlying device. When developing the location service, we can utilize Android's network location provider and/or GPS to acquire the user's location. They use different ways to obtain location data which differ in accuracy, minimum notification interval, time to first fix (TTFF), and battery consumption. The way to receive location information in Android is to register a location listener with the location manager. At registration, the location provider, important parameters for updating the location—frequency and moving distance, have to be specified.

GPS provides data with high accuracy: Less than 100 m is the official number, but real accuracy is often around 2 m - 20 m [49]. Once it is connected to satellites it can provide location updates in intervals as short as a few seconds. However, one drawback of GPS is the initial connection to the satellites - Time to First Fix (TTFF) - can be quite slow.

GPS traditionally needs to connect to at least three satellites for a 2D location, and initial transmission of so-called Ephemeris data takes a minimum of 30 seconds for a satellite. Once the Ephemeris data has been transmitted and connection to the satellite is made, location updates are very fast. To speed up the TTFF, manufacturers have come up with a new way to deliver the initial Ephemeris data to the GPS. They use an Internet connection to deliver the data much faster than the transmission from the satellite. Android uses this so-called Assisted GPS (A-GPS) to speed up TTFF to 5 - 15 seconds, but only if you have an Internet connection. Another drawback is that the battery consumption is quite high, and continued use of the GPS will drain the battery which limits the type of applications that can benefit from using the GPS. Last but not least, since GPS uses a radio signal, solid objects will obstruct the signal. In practice, this means the GPS will not work in most buildings.

The network location provider uses both WiFi hotspots and cell towers known to the Android device to approximate a location of a user. When the location provider is polled, the IDs of the WiFi hotspots and cell towers in the area are sent via Internet to the Google location server, a database with location information on WiFi hotspots and cell towers. Google location server returns an approximate location of the user. The data comes from a combination of cell and WiFi hotspot information, and can only be obtained when the device can access the Internet to query the Google location server. Accuracy of the data is specified somewhere between 100 m - 1000 m depending on the information available. WiFi hotspots allow an accuracy of 100 m - 500 m, while cell towers only allow for an accuracy of more than 500 m. This means the network provider will be very inaccurate in areas without WiFi hotspots. Conversely, the more hotspots and cell towers are in the area, the greater the accuracy is. Faster than the GPS, the network provider can give first location results within seconds. In addition, network location provider works indoors and outdoors and uses less battery power.

Acquiring the location data is handy. However, the “not easy” part for the location service is defining a model for the best performance. Figure 4.3 shows a timeline representing the window when the user begins and stops listening for location updates in our mobile application.

Under the condition of offering the user the information of others around her are also using CredFinder, the location service should be as power efficient as possible. The size of the listening window is minimized to the moment a user is watching that who are around

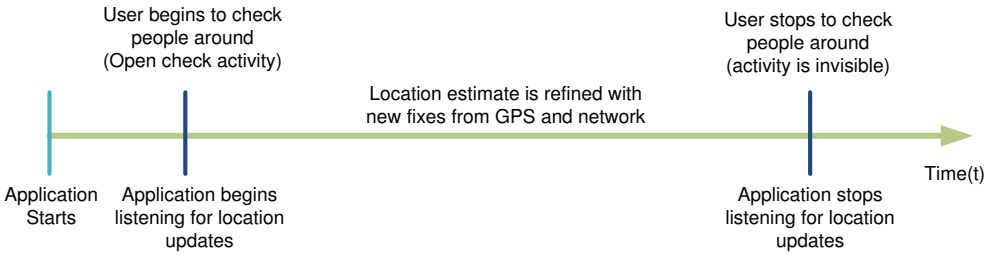

Fig. 4.3 A timeline representing the location update window

her to save the device battery. We use both GPS and the network location providers, and we neither set the minimal update distance nor reduce the update frequently so as to give the best experience to the user. To our understanding, these settings won't cost too much energy since the user usually stays a short time on "people around you" screen as long as she can get the information in time. The flow chart 4.4 demonstrates the process of maintaining a current best location estimate [47], where CL is current location obtained and BL is the best location stored so far. If this program returns true, that means CL is a better estimate than BL and otherwise not.

4.2.4 Local Managers

Building on top of the fundamental modules, there are four managers, the location service manager, the token exchange manager, the setup manager and the token transmission manager, implementing all the operations of our application. Logically, we categorize these operations into two phases, the server connection phase and the token exchange phase. Therefore, we describe how our application works according to these two phases.

Server Connection Phase

The procedure of the server connection phase is based on Section 3.3.2 step 1 to step 5 of the protocols with more practical concern. Every time the user starts our application, the setup manager will check whether there is login information stored. If not, the user will be prompted to log in. Afterwards, the token transmission manager checks whether there is

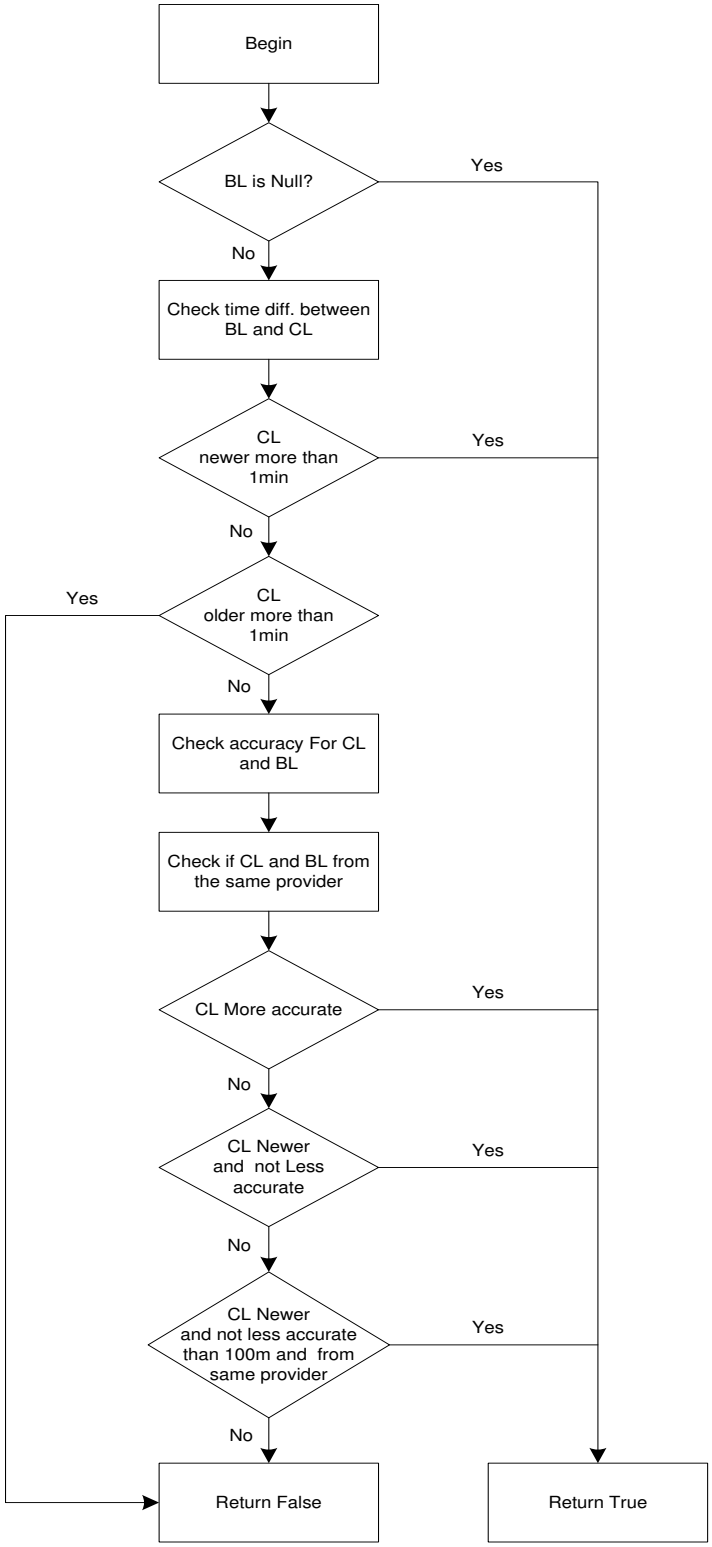

Fig. 4.4 Process of maintaining a current best location estimate

token left to be transmitted. If not, it will ask the validation server for another 100 tokens, otherwise do nothing. We set the number 100 to balance the device storage capacity and cost of asking for the new tokens. All the operations above are running automatically without bothering the user. And certainly, all the potential communication is running in a secure channel.

For our customized protocol between a mobile device and the validation server we define the following message format:

$$[START]command \sim parameter\_1, parameter\_2, ..., parameter\_N[END] \tag{4.1}$$

where "command" represents the word we use to distinguish each specific step and "parameter" is the optional parameters for this command.

Token Exchange Phase

This phase achieves step 6 to step 11 in the protocols. When people meet, the location service manager takes the responsibility to check the candidates around the user and lists them on the user's screen. To know the people around, each user needs to update her or his location according to the location service module. Every time the validation server receives a new location data from the user, it will calculate the people around the user and send the data back.

The sending part of the token exchange manager would help the user retrieve a token from token storage. After adding the system time and location obtained from the location service module, the token exchange manager sends it to the target person's device. Once a token is sent, the manager will check if it requires to ask for new tokens from the validation server. The receiving part of the manager helps the user take the token and uploads it to the validation server by invoking the token transmission manager.

For our protocol between mobile devices we define our own message format:

$$[START]USERID@UserInfo[END] \tag{4.2}$$

where "USERID" is the user's unique Facebook id and "UserInfo" is "token:time:location". We do not transfer the user's name since using USERID it can be easily retrieved.

4.2.5 User Interface

In the Android platform, "View" objects are the basic units of user interface expression. The "View" class serves as the base for subclasses called "widgets", which offer fully implemented UI objects, like text fields and buttons. The "ViewGroup" class serves as the base for subclasses called "layouts", which offer different kinds of layout architecture, like linear, tabular and relative. In particular, layout is the architecture for the user interface in an Android activity. It defines the layout structure and holds all the elements that appear to the user. We can declare the layout in two ways: Declare UI elements in XML or instantiate layout elements at runtime. In CredFinder mobile application, we define the following UIs.

- *Home screen*: The application begins here, which integrates "Facebook Login", "People Around" and "Ping" buttons.
- *People Around screens*: List the other users around and provide a button for the user to open the "Send Token" screen. Besides the automatic location update and information retrieval, we also give the user an option to force to update any time she wants.
- *Send Token(Ping) screens*: Screens for the user to scan Bluetooth devices, send and receive tokens from selected devices.
- *Notifications screens*: Notify the user for various activities such as token sent/received, Bluetooth device connected/disconnected and other application status changes.

It is worth noting that to list the people around a particular user, we populate a "View" group with some information that can't be hard-coded, therefore, we must bind the "View" to an external source of data which in our case is the data of other users around. Additionally, we have been consistently trying to provide a user-friendly interface. For example, to refuse replying a token, the user can either click the button or shake the phone.

4.3 Validation Server

The validation server is the core processing node supporting the mobile application as well as the OSN application server. Components of the validation server are revealed in Figure 4.5.

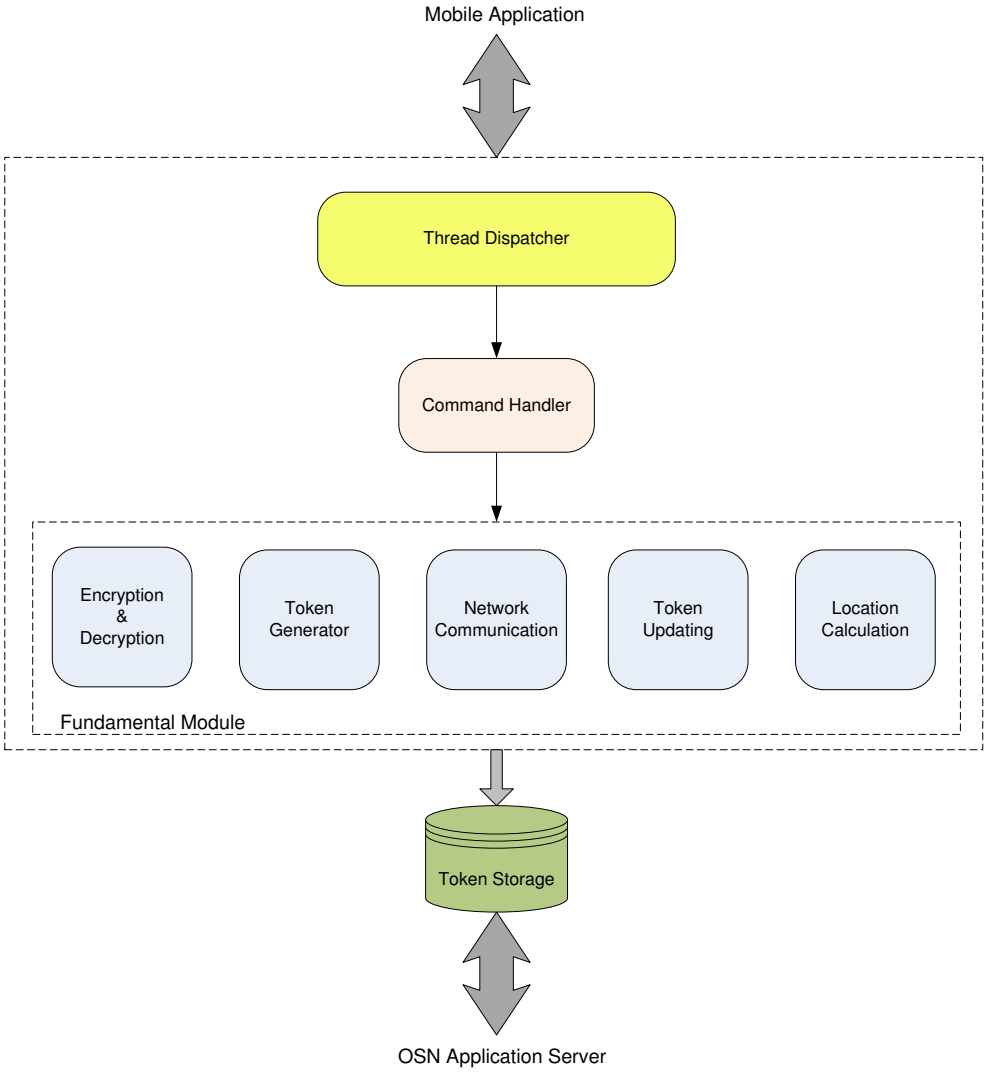

Fig. 4.5 Components of the validation server

Network Communication: Supported by Java Socket API, this module provides server side function for the communication with the user's Android application.

Encryption and Decryption: This module plays the same role as in the Android application.

Token Generator: Once the user's mobile device detects that there is no more token left to send, it will automatically ask for new tokens. When the validation server receives this request, 100 identity tokens will be generated by consecutively hashing a secret number using MD5 in a separate thread. Attached with user's id, these tokens then are sent to the user's mobile device as well as saved in the user's personal token file for further usage by the OSN application server.

Token Updating: At the moment one mobile device receives token from another device, it should upload this token to the validation server which will add this token to this user's token received list. This list is in fact the user's friend list in the real world and by gathering all users' friend lists, we can ultimately build an identity validation network.

Location Calculation: One of the function in the Android application is checking out other users nearby. This is implemented in the server's location calculation module. 1) The Android application delivers the current best location estimate to the validation server. 2) The server removes the corresponding previous location stored in the memory and puts the new location in. 3) The server computes the locations near the target location. 4) The server sends the associated user IDs and names found by location calculation to the user. For performance consideration, we maintain a hash table with key "user ID" and value "location" in memory as the data structure supporting the computation.

Moreover, we apply the 'haversine' formula [50] to calculate the great-circle distance, the shortest distance over the earth's surface, between two points in the form of longitude and latitude. This formula listed below is on the basis of a spherical earth (ignoring ellipsoidal effects), which is accurate enough for most purposes.

$$a = sin^2(\Delta lat/2) + cos(lat_1).cos(lat_2).sin^2(\Delta long/2) \tag{4.3}$$

$$c = 2.atan2(\sqrt{a}, \sqrt{1-a}) \tag{4.4}$$

$$d = R.c \tag{4.5}$$

where R is earth's radius (mean radius = 6,371km), lat is latitude and long is longitude.

Note that angles need to be in radians to pass to trigonometric functions. The haversine formula remains particularly well-conditioned for numerical computation even at small distances.

Thread Dispatcher and Command Handler: In general, the validation server is always waiting for the connection from the mobile devices. When a particular device connects to the server, it will dispatch a new thread to take care of this device. In this work thread, the command handler takes the responsibility for capturing the command from the mobile device and invoking the corresponding modules to carry out a specific task. After a task finishes, the command handler would gain back control waiting for the next command until the mobile application closes the connection.

Token Storage: The validation server creates two files for each CredFinder user, one for storing sent tokens and the other for storing received tokens. Those received token files construct the users' friend lists and then contribute to the identity validation network graph, while sent token files can help the system to check whether a received token is valid (This is implemented by comparing a received token to transmitter's sent token file). From the perspective of the OSN application server, it will access these token files when a user makes some request like trying to add someone as a friend on Facebook.

4.4 OSN Application Server

4.4.1 Web Service Overview

The OSN application web server powers a website where the user's validation happens. Focusing on Facebook, this OSN application website provides additional information such as where and when two people met in real life to help Facebook users validate the persons who make the online "add friend" requests. When a particular user is intended to validate the person she has met no matter what scenario mentioned in Section 3.1 it falls in, she should open our website and log in with her Facebook account. We need the user to grant us permission to retrieve some of the information of her Facebook profile for the first time. Afterwards, she no longer needs to fill out other registration form or remember other username and password to use our site. As long as the user has signed into Facebook, she automatically signs into our website as well. The user would be shown her token exchange history with the person requesting the friendship, which could greatly help the user verify

the people in an OSN and make the appropriate decision. Moreover, beside the stand-alone website, we also create a Facebook canvas page integrated with Facebook application platform to make convenience for users.

4.4.2 Facebook user ID

In our identity validation system, it is the user's Facebook ID that interlaces the mobile application, the validation server and the OSN application server and threads them together to offer an integrated service to every user.

- In the mobile application, users log in using their Facebook IDs.
- In the validation server, identity tokens are stored and updated by users' Facebook IDs.
- In the OSN application server, people log in with Facebook IDs and tokens are presented to users according to their IDs.

It should be noted that the validation server obtains the user's Facebook id passed by the mobile application.

As in the mobile application, the application server employs Facebook authentication SDK of JavaScript as well as PHP to implement the function. Facebook platform uses OAuth 2.0 for authentication and authorization. Open source JavaScript SDK provided by Facebook is the simplest way to use for login. On the other hand, the PHP SDK provides a rich set of server-side functionality for accessing Facebooks server-side API calls. These include all of the features of the graph API, FQL, and the deprecated REST API. The PHP SDK is typically used to perform operations as an application administrator, but can also be used to perform operations on behalf of the current session user. To power our application, we use PHP SDK in conjunction with the JavaScript SDK to provide seamless session management across both the client- and server-sides of the application.

4.4.3 Website Building and Hosting

We have built our website from scratch on Apache web server using our own Linux machine. However, to handle the potential increasing requirement of the user and to develop our server in a limited time, we choose to apply Drupal, a content management system (CMS), to construct our website. It cannot be denied that other CMSs like Wordpress and Django

are also good choices. However, considering the programming flexibility and integration with Facebook PHP API, Drupal becomes our primary choice.

Drupal is a free and open-source content management system and content management framework powering millions of websites written in PHP and distributed under the GNU General Public License. It is used as a back-end system for at least 1.5% of all websites worldwide ranging from personal blogs to corporate, political, and government sites including whitehouse.gov and data.gov.uk [51]. It is also used for knowledge management and business collaboration. The standard release of Drupal, known as Drupal core, contains basic features common to CMSs. These include user account registration and maintenance, menu management, RSS-feeds, page layout customization, and system administration. The Drupal core installation can be used as a brochureware website, an Internet forum, or a community website providing for user-generated content. As of August 2011 there are more than 11,000 free community-contributed add-ons, known as modules, available to alter and extend Drupal's core capabilities and add new features or customize its behavior and appearance. Because of this plug-in extensibility and modular design, Drupal is sometimes described as a content management framework. As a matter of fact, we could directly use "Facebook connect" module to satisfy our login requirement. It is also described as a web application framework, as it meets the generally accepted feature requirements for such frameworks.

Drupal is able to run on any computing platform that supports both a web server capable of running PHP (We use Apache2) and a database (we use MySQL) to store content and settings. Currently, we host the webserver using our own machine. We are willing to migrate the host to Amazon Elastic Compute Cloud if needed.

Chapter 5

Results and Performance Analysis

Having implemented the system we can now consider its practicality. In this chapter, I first present the implementation results which focus on the mobile application and the web service. Then various system quotas such as connection setup time, location service accuracy and battery life are tested and measured. The source code of the project can be reached at git@github.com:sssyyw/CredFinder.git. Anyone interested in CredFinder is welcomed to work on it.

5.1 Implementation Environment

We list our test environment of both hardware and software as follows. The performance may vary depending on the platform used.

- *Mobile device*: Samsung Galaxy Ace S5830, with 800 MHz ARM 11 CPU, 278 MB RAM, 158 MB internal storage and Android 2.3.4 system. Bluetooth v2.1, A-GPS and Wifi are built in.
- *Validation server*: Intel P8400 2.26GHz duo CPU, 2GB RAM and 500GB 7200 rpm hard drive. Ubuntu 10.04 and Jdk 1.6-34 are installed.
- *OSN application server*: Intel P8400 2.26GHz duo CPU, 2GB RAM and 500GB 7200 rpm hard drive. Ubuntu 10.04, Apache2, MySQL and PHP5 are integrated.

5.2 Results

5.2.1 Android Application

Although a lot of details can be presented in this section, we focus on the main features of our Android application.

Home Screen and Login

To keep our application simple and clear, as shown in Figure 5.1, we only put three buttons in the home screen as described in Section 4.2.5. Hopefully, the application can explain itself. To use our service, typically the first thing the user should do is login the system with, in our prototype, Facebook account. If the authentication succeeds, user can start playing with the application. Figure 5.2 unveils that the author logged on the system with his Facebook account.

Fig. 5.1 Application Home screen

Fig. 5.2 Logged in screen

Location Service

"People Around" button brings the user to the location service where the user can look up who else using CredFinder is nearby. Once the button is clicked, the service screen jumps in and the location service starts running. User can let the service run automatically or manually by pressing "update now" and then a list of other users will display on the screen. For example, in Figure 5.3, Steve Gatesburg and Yinan Zhang is near our tested user Yiwei Shi. If wanted, the user can directly begin exchanging the token with any one in the list by clicking "Ping" button at the bottom of the screen, which will bring our user to another screen to choose the associated device name for Bluetooth connection. After user leaves this screen, the location service will be terminated accordingly.

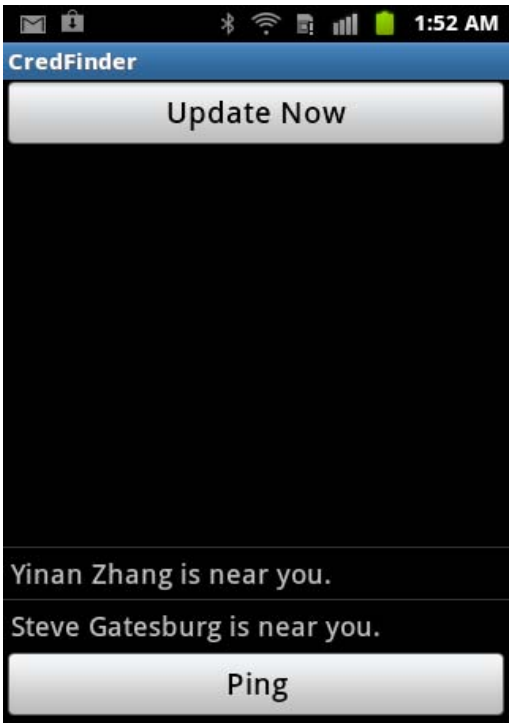

Fig. 5.3 List other users around

Token Exchange

Through the button "Ping" in either home screen or the location service screen, the user will go to the token exchange interface. After the Bluetooth scanning, pairing and connecting

preparation introduced in Section 4.2.3, the application is ready to send and receive the identity tokens. For instance, if a user Yinan sends a token to another user Yiwei, on Yiwei's screen, "Token received from Yinan using device GT-S5830" would be printed out at same time "Reply" would be prompted (See Figure 5.4). If Yiwei agrees to reply, after sending the token successfully, "Token sent to device GT-S5830" will emerge as shown in Figure 5.5. Thus, a local token exchange process is completed and the received token will be uploaded to the validation server afterward.

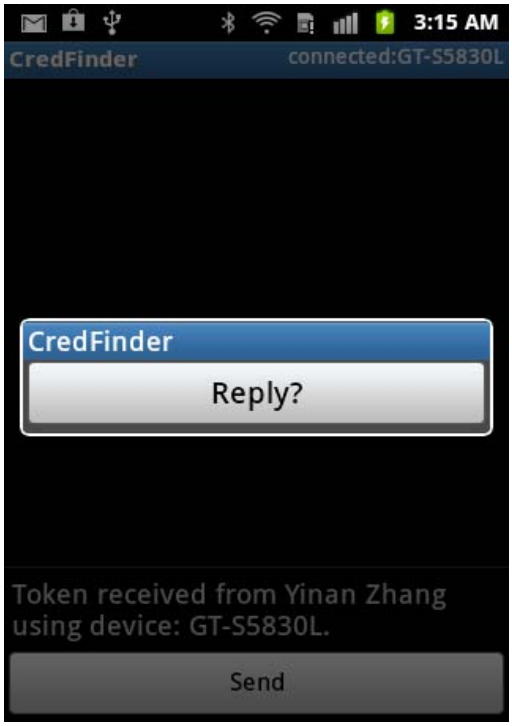

Fig. 5.4 Token exchange 1

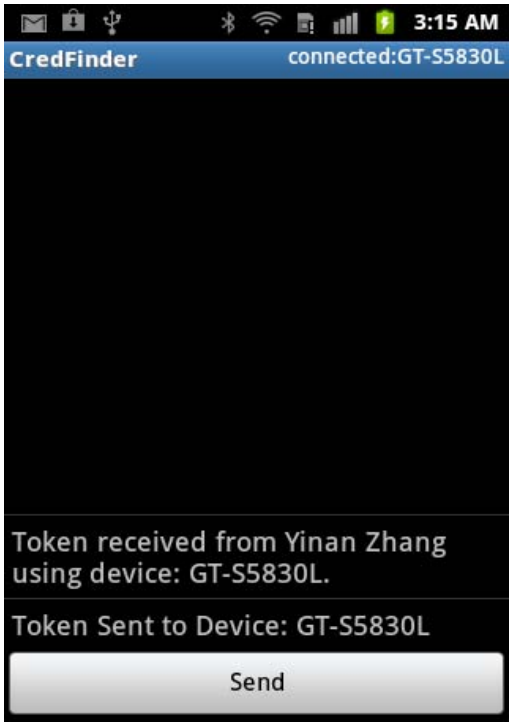

Fig. 5.5 Token exchange 2

5.2.2 Validation Server

The validation server takes the responsibility for building our validation network and supporting the service in both the mobile application and the OSN application website. We applied the unit tests as well as the ones integrated with the other two system components to ensure it works well.

5.2.3 OSN Application Server

The mobile application is the tool for the user to exchange the credentials, while the OSN application website is a window for the user to see what she has. Figure 5.6 shows the home page of our OSN application website. With the user's Facebook account, one could check out who has exchanged tokens with her before. The time and location of the given token exchanges are also provided. Moreover, for Scenario 2, the user could explore the validation networking connection and examine how many people link her and the target person together.

Fig. 5.6 OSN application website

5.3 Performance Analysis

In this section, the performance of our mobile application, particularly the performance bottleneck for such a prototypical system, is emphasized.

5.3.1 Bluetooth Connection Setup Time

Our mobile application utilizes Bluetooth to exchange tokens between each other, which is also a key feature manipulated by the user. Therefore, the performance of the token exchange is crucial to the application, whereas Bluetooth connection setup time is a significant index for measuring it. We did the setup time test in a corridor in the FDA building of McGill. The test result is shown in Figure 5.7.

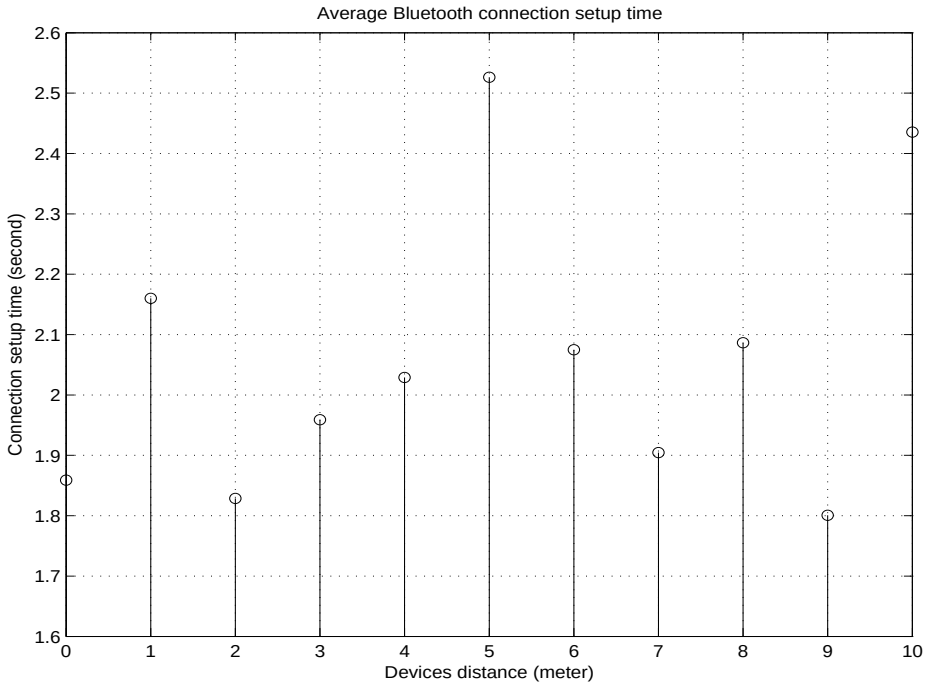

Fig. 5.7 Average Bluetooth connection setup time

The horizontal axis represents the distance between two devices, while the vertical axis represents the connection setup time. We chose test distances from 0 to 10 meters at every integer number point. At each distance, we collected 10 samples and presented the mean of this 10 setup time on the graph. It is worth noting that the connection setup time measured here is round-trip time. Interestingly, out of our expectation, the average setup time did not increase as the distance grew. In fact, it shows a random distribution that the setup time is relatively short at 0m, 2m, 7m and 9m and a little bit longer at other distances. We further dug into the standard deviation of the 10 samples in each test distance, as illustrated in Figure 5.8, which indicates that the setup time does not

vary dramatically since the standard deviation is between 0.3s to 1.3s approximately. The reason why it takes about 2 seconds to set up the Bluetooth connection is as follows. In order for a device to connect to another Bluetooth device it uses the page scan channel. An unconnected Bluetooth device must periodically enter the page scan state; in this state, the device activates its receiver and listens for a master device that might be trying to page it. If the paging device does not know the phase of the target device's page scan channel, it therefore does not know the current hop frequency of the target device. The paging device transmits page requests on each of the page scan hop frequencies and listens for a page response. A device operates in one of three page scan repetition modes for a master paging it, device listening continuously, device listening at least every 1.28 seconds, and device listening at least every 2.56 seconds [52]. Therefore, depending on the different scan repetition modes the device is in, the time of the page scanning plus the partial time of its interval could be a couple of seconds. Furthermore, the paging device may have some knowledge of the target device's Bluetooth clock in which case it is able to predict the phase of the target device's page scan channel. It may use this information to optimize the synchronization of the paging and page scanning process so as to speed up the formation of the connection. That explains why in our experiment, a few setup time were as short as 400ms.

We assume the token exchange typically happens within 10 meters according to the property of CredFinder. However, through the experiments, we found that the average Bluetooth connection setup time did not decline significantly even the test distance extended up to 25 meters, given that there was no obstacle between the users. We have also noticed that the connection failure emerged as the distance became longer than 10 meters. For the scenario that there were insurmountable obstacles like a door between the users, the connection failure rate soared at any given distance.

It should be noticed that the Bluetooth turning on or turning off time is typically over 3 seconds on Android devices. But since the switching is not frequently operated by the user and also those time cannot be optimized by the application, we do not bother to put emphasis on that.

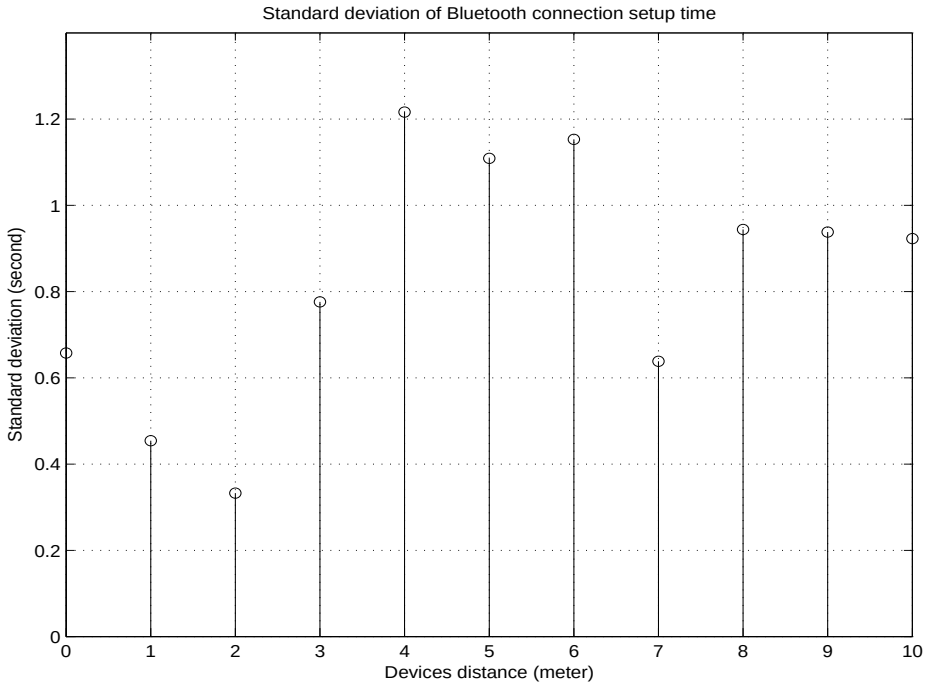

Fig. 5.8 Standard deviation of Bluetooth connection setup time

5.3.2 Device and Validation Server Connection Setup Time

The mobile devices and the validation server communicate frequently in our system, therefore the connection setup time between them should also be measured. In our experiment, the average connection setup time was about 0.1 seconds. Though this result can be varied depending on different network environment characterized by throughput, congestion and so forth, it indicates that this setup time is quick enough to support the mobile device application.

5.3.3 Location Service Accuracy

We have used both the network provider and GPS to retrieve the user's location in our application, thus we measured the accuracy of them separately.

Network Provider

The location accuracy by the network provider was measured around the McGill campus. The range of the accuracy was from 40 meters to 75 meters depending on different spots.

This accuracy was achieved due to the fact that there are lots of WiFi hotspots assisting the positioning within the campus.

GPS

Similarly measured around McGill campus, the range of the GPS accuracy was from 4 meters to 48 meters. It was more accurate than the one from the network provider outside with the typical accuracy of around 6 meters to 12 meters. However, in the building it was hard to receive the GPS signal. Additionally, we observed negative correlation between the accuracy of network provider and GPS. That is when the accuracy from GPS went up from indoor to outdoor, the accuracy from network provider went down.

In our application, we combine these two ways together to provide the location service. Since either GPS or the network provider is in a good accuracy situation at a given time, our strategy can satisfy the user's requirement.

5.3.4 Battery Life

Battery usage is another major factor to leverage the mobile application. Both the Bluetooth and the location service are regarded as important battery drain sources. In this experiment, the capacity of battery was 1350 mAh.

Location Service and Battery Life

The location service provides the user the ability to check out who else is around her for the potential token exchange, which is implemented by continuously pushing the user's location to the validation server and pulling out the information of other users nearby. The battery usage for the location service changes dramatically. It depends on where the user is and how fast the user is moving since we set the minimum updating distance and apply an algorithm to obtain the best location estimate. Therefore, we measured the battery usage using the network provider and GPS respectively, and assumed that the minimum updating distance and minimum updating interval were both 0 for the measuring purpose.

For the network provided location service, it drained the whole battery in 33 hours 15 minutes, which was a little bit over 3% per hour. This result was acquired on an average 60 meters location accuracy and around 45 seconds per update. When it came to the GPS, it consumed the battery life in 3 hours 30 minutes, which was acquired on an average 8 meters

accuracy and around 3.7 seconds per update. That was because the updating frequency of GPS was much greater than that of network provider. In practice, by setting up a 10 meters minimum updating distance, the GPS updating frequency would be largely reduced. Additionally, given that the location service is automatically terminated once the user leaves the "People Around" activity, the GPS battery consumption is limited. Furthermore, as seen in Section 4.2.3, our application combines these two location providers together to offer the best calculation result to the users.

Bluetooth and Battery Life

For Bluetooth, the biggest contributor to the battery usage is Bluetooth scanning. To measure it, we simply ran our program and scanned 12 seconds for Bluetooth connections every 8 seconds. This drained the battery in 22 hours, or in other words, the Bluetooth could ceaselessly scan for almost 15 hours. It cannot be neglected that Bluetooth enabled discovery and token exchange also share the battery usage. However, according to our tests, they only have limited impact. Bluetooth discovery mode consumed 2% battery usage per hour, which was almost like the Bluetooth standby power consumption. Token exchange also drains the battery life. In our measurement, considering that 100 token exchanges per day can satisfy most users' requirements, we did 100 times of the Bluetooth connection setup and the token exchange between two devices one meter away from each other. The result shown that this cost 2% of the battery. Due to our automatic Bluetooth shutting down scheme and the limited usage for the user to exchange tokens per day, we believe that the Bluetooth battery usage is fairly acceptable.

Chapter 6

Security Concerns

As an identity validation system, CredFinder's own security is a significant issue. In this chapter, I discuss the common threats we anticipate in the CredFinder system and the counter measures we have. Anyone can become a valid user of CredFinder including the adversary. The architecture of our system minimizes the chances for the attacker to gain others' validation data. But we do need to pay attention to the various ways to subvert our system. In general, the security threats are categorized into two categories, the attacker with no stolen devices and the attacker with one stolen device. The cases which the attacker with more than one stolen device are more complicated and can be derived from the one stolen device case. Concentrating on the attacks which may take advantage of our unique system architecture, I intentionally bypass some common attacks including many web service attacks.

6.1 Attacker With No Stolen Devices

6.1.1 Wireless Communication

For the mobile application, one important security concern is wireless communication. For the communication between the mobile device and the validation server, we apply Diffie - Hellman key exchange to build a secure channel for data transmission. For the token exchange between mobile devices, because the Bluetooth's effective range is typically less than 10 meters, the attacker has to stay close to the users in order to intrude the communication. Even if there exists bold attackers, the Bluetooth's protection mechanism is

strong enough to keep them away from attacking. In order to provide usage protection and information confidentiality, Bluetooth provides security measures both at the application layer and the link layer [52]. These measures are designed to be appropriate for a peer-to-peer environment. Four different entities are used for maintaining security: a Bluetooth device address, two secret keys(one for authentication and the other for encryption), and a pseudo-random number that shall be regenerated for each new transaction.

6.1.2 Attacker as a Legal CredFinder User

CredFinder is open to every OSN user including the malicious one. The malicious user can also install our mobile application on his own phone and try to tempt his target to grant him a token. Again, since the token exchange process is designed to happen in the face to face scenario, we have good reasons to think that the users would refuse granting their tokens to unknown persons so that the attacker has less chances to intrude the users' token exchanged friend lists.

6.1.3 Group of Attackers as Legal Users

Another concern is that, though rarely happening, it is possible that there is a group of attackers collaborating together to intrude the system. That is to say a couple of attackers install our mobile application and exchange tokens with each other. Fortunately, it takes no effect to our system except for Scenario 3. In this scenario, a person who is an adversary wants to add Alice as a friend, and what he can provide from the CredFinder is some credentials from his "friends" which are also adversaries. The adversary is hoping that the victim could accept the request. It is dangerous if the victim accepts the request in this scenario, however, our system will show all the credentials' owners name to the victim, if Alice does not know any of them, we believe that Alice will decline this "friendly" request.

6.1.4 Location Service Attack

Since GPS is a one way system (i.e. the user's device only receives satellite signals and computes its location itself), it can receive fake location information. Preventing this requires a trusted or tamper-proof GPS device. Hacking such a trusted system would present a difficult challenge, since it requires generating fake GPS satellite signals from at least 3 sources, which is difficult but not impossible for a resourceful attacker. This attack

can be prevented by using the anti-spoofing information included in the GPS signal. This information is ignored by civilian users, and requires an encryption key that is only available to the defence establishment. Since there is secure bidirectional communication, the cell tower determines the user's position with high confidence. Fake location requires cloning the phone's SIM card which is rather difficult. Even if our location service is attacked, it does not really harm our users because the location service only tells the user who else is around. The token exchange process must be interfered by the user.

6.2 Attacker With One Stolen Device

CredFinder follows the familiar "you hold this token so it is you" model, therefore another threat exists pertaining to the loss or theft of the user's mobile device.

6.2.1 Token Granting Abuse

CredFinder mobile application does not require username/password combination every time the user accesses it, thus once the attacker has stolen a user Alice's mobile device, he can send Alice's identity tokens to himself or any other user. But this will merely raise his reputation in the identity validation network by one token. More importantly, there are two possibilities afterwards. If Alice figures out that she has lost the phone in time, she could easily find out those bogus tokens sent by the attacker by checking the token sending time later on in the validation server. Then Alice would make these tokens deprecated. If Alice is not aware that she has lost the phone (probably because the attacker returns the phone after the token exchange manipulation which more rarely happens), she still has the chance to detect the abnormal token exchange by scrutinizing where and when the exchange happened, assuming that generally the attacker does not take the risk to stay with Alice and use her phone. After all, it is very dangerous for attacker to do the token exchange and return the phone in that Alice may easily know who did this.

6.2.2 Peep the Token

A straightforward behavior an attacker will do when obtaining Alice's phone is peeping her tokens and then using them for himself. However, we securely store the user's credentials so that it will difficult for attacker to obtain them. Even if an attacker acquires the token,

the public-key encryption insures that the user's Facebook id will not be leaked for some malicious usage. Moreover, since attacker cannot know the victim's id, he cannot intrude CredFinder by pretending to be someone else.

6.2.3 Add Friend Using Stolen Device

We cannot neglect that most users who use our application may install Facebook Android application since it is very popular and our application is based on Facebook. So when an attacker steals such a phone from Alice, he can directly use her Facebook Android application to add her as a friend to avoid being checked by CredFinder. We are not able to manipulate other applications and hence handle this problem except hoping that Alice would check out Facebook and CredFinder later when she finds out this "friend" is kind of suspicious.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis we have designed three protocols of identity validation for social networks based on the mobile device application. There are three major components, mobile device application, validation server and OSN application server. We developed an Android application on commodity cell phones, constructed a Java based back-end validation server and built a web server on LAMP architecture to implement a prototypical system. One of the core concepts in our system is "mobile device as identity token". People confirm their meeting in real life by exchanging their identity tokens carried on mobile devices via Bluetooth. These token transmissions will be presented to the users later and help them validate people in OSNs like Facebook. We tested the availability for every system function. We measured the Bluetooth connection setup time, location service accuracy, mobile device battery life, etc. to ensure CredFinder's quality of service. Besides, user experience is also considered when developing the whole system. Finally, we investigated various threats against the user availability. With the encrypted tokens and a safe communication channel, the attacker cannot crack our system and then launch a potential identity theft attack. Even if the user loses her mobile device, with the data of where and when the token exchange happened, there is still a reasonable chance that the user is not being harmed.

CredFinder makes a major improvement over the current state of anti-identity theft attack system for online social networks. It abandons the complicated theft detection algorithms which are not fully verified in the real OSNs. To the best of our knowledge, CredFinder is the first mobile device based practical system against social network identity

theft attacks. The validation strategy in our system gives the ordinary users the power to control their own identities tokens. In addition, using their mobile devices, users can build a big trustable validation graph. We also can use this valuable real world network information to back up other user related services, for instance, creating a more trustful social authentication system.

It cannot be denied that there is one drawback of CredFinder. CredFinder is still a passive system, which means neither the system service nor the user would aggressively detect the adversaries. The validation happens only when the adversaries are trying to start the attacks. This drawback is caused by the intrinsic property of our system that CredFinder is a practical system built by every user, therefore we have to sacrifice something to maximize the benefits.

7.2 Future Work

So far, CredFinder has only gone through the internal test and measurement. In the future, we plan to make public test on selected students in McGill computer science department. This experiment would give us the practical data of how well CredFinder works. And the feedback of the users would be very helpful for the following research.

Our current implementation is deployed on Android mobile operating system and Facebook. To enlarge the system accessability, our system should also be compatible with other major OSNs like Linkedin or Google+. Moreover, our mobile application should be extended to iOS and Windows mobile phone since millions of users are using them. These expansions could really make CredFinder a cross-platform system which serves more people.

For a practical system, the CredFinder's user experience has a lot of room to improve. For example, an optimization of the token exchange triggering algorithm need to be considered so that the user can have the one-click token exchange experience. Moreover, more friendly and personalized user interface of the mobile application as well as the CredFinder website is bound to be explored. We also consider allowing the user to exchange their tokens through SMS or data communication in some circumstance to assist the user who is not able to send the token promptly due to some reasons.

References

[1] "Wikipedia, social network service," http://en.wikipedia.org/, 2011.

[2] "Facebook statistics," http://www.facebook.com/press/info.php?statistics, 2011.

[3] "Linkedin press center," http://press.linkedin.com/about, 2011.

[4] "New myspace and facebook worm target social networks," http://www.darknet.org.uk/2008/08/new-myspace-and-facebook-worm-target-social-networks, 2008.

[5] A. A. Hasib, "Threats of online social networks," *IJCSNS International Journal of Computer Science and Network Security*, vol. 9, no. 11, pp. 288–293, November 2009.

[6] L. Bilge, T. Strufe, D. Balzarotti, and E. Kirda, "All your contacts are belong to us: automated identity theft attacks on social networks," in *in Proceedings of the 18th international conference on World Wide Web, Madrid, Spain*, pp. 551–560, 2009.

[7] L. Jin, H. Takabi, and J. B. Joshi, "Towards active detection of identity clone attacks on online social networks," in *ACM Conference on Data and Application Security and Privacy, CODASPY'11, San Antonio, Texas*, pp. 27–38, February 21C23, 2011.

[8] "Wikipedia, online identity," http://en.wikipedia.org/wiki/Online_identity, 2011.

[9] Y. Xu, "Resisting whitewashing: A comparative study of fixed identity, pseudonym and social identity," Master's thesis, School of Computer Science, McGill University, Montreal, Canada, Nov. 2011.

[10] D. Chaum, "Security without identification: Transaction systems to make big brother obsolete," *Communications of the ACM*, vol. 28, no. 10, pp. 1030–1044, 1985.

[11] Wiszniewski, Dorian, and R. Coyne, *Building Virtual Communities: Learning and Change in Cyberspace*. Cambridge. Cambridge University Press, 2002.

[12] F.-Y. W. et al., "A study of the human flesh search engine: Crowd-powered expansion of online knowledge," *Computer*, vol. 43, no. 8, pp. 45–53, Aug. 2010.

[13] D. Recordon and D. Reed, "Openid 2.0: a platform for user-centric identity management," in *in Proceedings of the second ACM workshop on Digital identity management, New York, NY*, pp. 11–16, 2006.

[14] B. Ford and J. Strauss, "An offline foundation for online accountable pseudonyms," in *in Proceedings of the 1st workshop on Social network systems, New York, NY*, pp. 31–36, 2008.

[15] M. Yokoo, Y. Sakurai, and S. Matsubara, "The effect of false-name bids in combinatorial auctions: new fraud in internet auctions," *Games and Economic Behavior*, vol. 46, no. 1, pp. 1030–1044, 1985.

[16] M. Maheswaran, B. Ali, H. Ozguven, and J. Lord, "Online identities and social networking," *Handbook of Social Network Technologies and Applications*, pp. 241–267, 2010.

[17] M. Subramani and B. Rajagopalan, "Knowledge-sharing and influence in online social networks via viral marketing," *Communications of the ACM*, vol. 46, no. 12, pp. 300–307, 2003.

[18] H. Choi, S. Kruk, S. Grzonkowski, K. Stankiewicz, B. Davis, and J. Breslin, *Trust Models for Community Aware Identity Management: Trust Management in Virtual Environment*. Icfai Books, June 2008.

[19] B. Jennings and A. Finkelstein, "Digital identity and reputation in the context of a bounded social ecosystem," in *in Business Process Management Workshops*, (Springer), pp. 687–697, 2009.

[20] J. Brainard, A. Juels, R. Rivest, M. Szydlo, and M. Yung, "Fourth-factor authentication: Somebody you know," in *in CCS'06: 13th ACM conference on Computer and communications security*, pp. 168–178, 2006.

[21] B. Soleymani and M. Maheswaran, "Social authentication protocol for mobile phones," in *in Computational Science and Engineering, 2009. CSE09, Vancouver, BC*, pp. 436–441, 2009.

[22] S. Sproule and N. Archer, "Defining identity theft," in *Eighth World Congress on the Management of eBusiness, WCMeB 2007, Toronto, Ont.*, pp. 20–30, Auguest, 2007.

[23] *Organisation for Economic Co-operation and Development, Online Identity Theft*. Paris : OECD, 2009.

[24] "Identity theft plagues canadians as online shopping grows," http://www.canada.com/story.html?id=b7f81191-421a-48f5-abc3-8b156c8f6fc2, 2008.

[25] "Bank of china phishing attack," http://blogs.mcafee.com/mcafee-labs/massive-online-bank-phishing-attacks-in-china, 2011.

[26] A. Marshall and B. Tompsett, "Identity theft in an online world," *Computer Law and Security Report*, vol. 21, no. 2, pp. 128–137, 2005.

[27] "Federal government's website to help safe, secure and responsible online," OnGuardOnline.gov, 2011.

[28] "Facebook development platform," http://developers.facebook.com/, 2011.

[29] A. Felt and D. Evans, "Privacy protection for social networking platforms," in *In Web 2.0 Security and Privacy Workshop*, 2008.

[30] K. Singh, S. Bhola, and W. Lee, "xbook: Redesigning privacy control in social networking platforms," in *In USENIX Security*, 2009.

[31] D. Rosenblum, "What anyone can know: The privacy risks of social networking sites," *IEEE Security and Privacy*, 2007.

[32] F. Stutzman, "An evaluation of identity-sharing behavior in social network communities," *International Digital and Media Arts Journal*, vol. 3, no. 1, pp. 10–18, 2006.

[33] J. He, W. Chu, and Z. Liu, "Inferring privacy information from social networks," in *Intelligence and Security Informatics*, pp. 154–165, 2006.

[34] H. Krasnova, O. Gunther, S. Spiekermann, and K. Koroleva, "Privacy concerns and identity in online social networks," *Identity in the Information Society*, vol. 2, no. 1, pp. 39–63, 2009.

[35] "Federal trade commission," http://www.ftc.gov/, 2011.

[36] T. N. Jagatic, N. A. Johnson, M. Jakobsson, and F. Menczer, "Social phishing," *Communications of the ACM*, vol. 50, no. 10, pp. 94–100, 2007.

[37] B. Markines, C. Cattuto, and F. Menczer, "Social spam detection," in *AIRWeb '09 Proceedings of the 5th International Workshop on Adversarial Information Retrieval on the Web, New York, NY*, pp. 41–48, 2009.

[38] J. Bonneau, J. Anderson, R. Anderson, and F. Stajano, "Eight friends are enough: social graph approximation via public listings," in *in Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pp. 13–18, 2009.

[39] "Sophos–facebook: The privacy challenge," http://www.sophos.com/en-us/security-news-trends/security-trends/facebook.aspx, 2011.

[40] R. Gross and A. Acquisti, "Information revelation and privacy in online social networks," in *WPES '05 Proceedings of the 2005 ACM workshop on Privacy in the electronic society, New York, NY*, pp. 71–80, 2005.

[41] "Sina weibo reaches 250 million users," http://news.ichinastock.com/2011/11/sina-weibo-reaches-250-million-users-sina-weibo-reaches-250-million-users/, 2011.

[42] "Fightidtheft," http://www.facebook.com/FightIDTheft, 2011.

[43] "Identity badge," http://apps.facebook.com/identity_badge, 2011.

[44] "mysafefriend," http://apps.facebook.com/mysafefriend, 2011.

[45] "Salman rushdie claims victory in facebook name battle," http://www.bbc.co.uk/news/uk-15733026, 2011.

[46] "Nfc forum," http://www.nfc-forum.org/aboutnfc/tech-enabler/, 2011.

[47] "Android developers," http://developer.android.com, 2011.

[48] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.

[49] "A good look at android location data," http://blog.shinetech.com/2011/10/14/a-good-look-at-android-location-data/, 2011.

[50] R. W. Sinnott, "Virtues of the haversine," *Sky and Telescope*, vol. 68, no. 2, p. 159, 1984.

[51] "Drupal - open source cms," http://drupal.org/, 2011.

[52] "Bluetooth specification, version 2.1 + edr," https://www.bluetooth.org/, 26 July 2007.