2009 International Conference on Computational Science and Engineering

Solutions to Security and Privacy Issues in Mobile Social Networking

Aaron Beach, Mike Gartrell, and Richard Han
{aaron.beach, mike.gartrell, richard.han}@colorado.edu
University of Colorado at Boulder

***Abstract*—Social network information is now being used in ways for which it may have not been originally intended. In particular, increased use of smartphones capable of running applications which access social network information enable applications to be aware of a user's location and preferences. However, current models for exchange of this information require users to compromise their privacy and security. We present several of these privacy and security issues, along with our design and implementation of solutions for these issues. Our work allows location-based services to query local mobile devices for users' social network information, without disclosing user identity or compromising users' privacy and security. We contend that it is important that such solutions be accepted as mobile social networks continue to grow exponentially.**

I. Introduction

Our focus is on security and privacy in location-aware mobile social network (LAMSN) systems. Online social networks are now used by hundreds of millions of people and have become a major platform for communication and interaction between users. This has brought a wealth of information to application developers who develop on top of these networks. Social relation and preference information allows for a unique breed of application that did not previously exist. Furthermore, social network information is now being correlated with users' physical locations, allowing information about users' preferences and social relationships to interact in real-time with their physical environment. This fusion of online social networks with real-world mobile computing has created a fast growing set of applications that have unique requirements and unique implications that are not yet fully understood. LAMSN systems such as WhozThat [1] and Serendipity [2] provide the infrastructure to leverage social networking context within a local physical proximity using mobile smartphones. However, such systems pay little heed to the security and privacy concerns associated with revealing one's personal social networking preferences and friendship information to the ubiquitous computing environment.

A. Our Contributions

We present significant security and privacy problems that are present in most existing mobile social network systems. Because these systems have not been designed with security and privacy in mind, these issues are unsurprising. Our assertion is that these security and privacy issues lead to unacceptable risks for users of mobile social network systems.

We make three main contributions in this paper.

1) We identify three classes of privacy and security problems associated with mobile social network systems: (1) direct anonymity issues, (2) indirect or K-anonymity issues, and (3) eavesdropping, spoofing, replay, and wormhole attacks. While these problems have been examined before in other contexts, we discuss how these problems present unique challenges in the context of mobile social network systems. We motivate the need for solutions to these problems.
2) We present a design for a system, called the identity server, that provides solutions for these security and privacy problems. The identity server adapts established privacy and security technologies to provide novel solutions to these problems within the context of mobile social network systems.
3) We describe our implementation of the identity server.

II. Background and Related Work

In this section we provide the reader with a short introduction to work in the area of mobile social networking and the technologies that have made it possible.

A. Mobile Computing

Smartphones now allow millions of people to be connected to the Internet all the time and support mature development environments for third-party application developers. This has put personal computing power in the pockets of users and at the same time, given them ubiquitous access to rich online social network information. In certain areas (such as college campuses) there are now high concentrations of active social network users with smartphones.

Recently there has been a dramatic rise in usage of smartphones, those phones capable of Internet access, wireless communication, and supporting development of third-party applications. This rise has been due largely to the iPhone and iPod Touch. In fact, according to Net Applications, Apple's handheld status symbol accounted for nearly two-thirds of all mobile web browsing traffic in April of 2009, almost eight times more than the nearest competitors [3]. This is amazing considering that less than a year before this the iPhone was not even the leading platform for mobile web traffic.

B. Social Networks

The growth of social networks has exploded over the last year. In particular, usage of Facebook has spread internationally and to users of a wide age range. According to

978-0-7695-3823-5/09 $26.00 © 2009 IEEE
DOI 10.1109/CSE.2009.243

IEEE computer society

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 1

TABLE I
SUMMARY OF SECURITY AND PRIVACY ISSUES FOR PEER-TO-PEER AND CLIENT-SERVER MOBILE SOCIAL NETWORK SYSTEMS

Security and privacy issue	Applies to peer-to-peer systems	Applies to client-server systems
Direct anonymity	Yes	Yes
Indirect or K-anonymity	Yes	Yes
Eavesdropping, spoofing, replay, and wormhole attacks	Yes	No

Facebook.com's statistics page, the site has over 200 million active users [4] [5], of which over 100 million log on everyday. To compare this with ComScore's global Internet usage statistics [6], this would imply that nearly 1 in 10 of all Internet users log on to Facebook everyday and that the active Facebook Internet population is larger than any single country's Internet population (China is the largest with 179.7 million Internet users [6]). Mobile users in particular are active Facebook users. According to Facebook statistics (March 2009) there are currently over 30 million active mobile users of Facebook, and those users are almost 50% more active on Facebook than non-mobile users.

C. Existing Mobile Social Network Applications

The unique mobile social network challenges described in this paper were discovered largely through the authors' prior work on WhozThat [1] and SocialAware [7]. Both of these were early systems to enable the creation of context-aware (location-aware) applications that exploit social network information found on existing online social networks such as Facebook.

Many applications have already taken rather simple and traditional approaches to integrating social network information with user location and context information. The most common form of application simply extends access to social networks to mobile phones or provides social network interfaces optimized for access from these mobile phones. For instance applications such as the iPhone or Blackberry Facebook applications[8] allow the user to natively interact with Facebook through his/her phone. Some work has taken a sensor network approach to mobile social networks, turning the phone into a sensor extension of the social network. CenceMe sends context information to the social network, e.g. the location of the user and perhaps context cues such as whether the user is talking [9]. This approach is rather unidirectional, focusing on enriching the social network (and its desktop applications) through the user's context. However, these applications do not consider that both the user's context and social network information can be more than the sum of their parts when integrated deeply on the user's mobile device. In contrast, the WhozThat system exploits mobile computing technology to import contextual information from social networking sites into the user's local physical environment. Serendipity [2] is a system similar to WhozThat that imports social context into the local context using mobile devices. However, Serendipity populates its own database of social context information rather then connecting with popular online social networking sites.

Commercial LAMSN services, such as Brightkite [10] and Loopt [11], provide some of the functionality found in WhozThat and SocialAware. However, like Serendipity, these services populate their own databases with social networking and context information, rather than leveraging popular online social networking sites such as Facebook. Also like Serendipity, these services do not consider the development of context-aware applications such as those enabled by WhozThat and SocialAware.

D. Privacy and Security

The work described in this paper draws on some previous privacy research in both location-based services and social networks [12] [13]. Previous work at Duke University [14] [15] has dealt with privacy and anonymity questions as they apply to sharing presence information with other users and matching users with a shared location and time. This prior work does not approach the same problem as addressed in this paper, however the mechanisms used in these papers may provide certain functions necessary to associate user preferences anonymously with user location for use in third-party applications. For instance, SmokeScreen [14] presents a protocol by which devices may broadcast identifiers that can be resolved to an identity through a trusted broker system. This identity could then be used to access personal information to drive third-party applications. Our work, however, differs in that it seeks to hide the user's identity while distributing certain personal information obtained from existing online social networks.

III. SECURITY AND PRIVACY PROBLEMS

Peer-to-peer mobile social network systems, like WhozThat and SocialAware, exchange users' social network identifiers between devices using short-range wireless technology such as Bluetooth. In contrast to these systems, a mobile device in client-server mobile social network systems, such as Brightkite and Loopt, notifies a centralized server about the current location of the device (available via GPS, cell-tower identification, or other mechanisms). By querying the server, mobile devices in these client-server systems can find nearby users, information about these nearby users, and other items of interest.

The following will discuss security and privacy problems associated with peer-to-peer and client-server mobile social network systems. Since there are differences between the peer-to-peer and client-server architectures, we will indicate which issues apply to a particular architecture. Table I summarizes the issues for each architecture.

A. Direct Anonymity Issues

The information exchange model of the mobile social network systems discussed previously provide little protection

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 2

for the user's privacy. These systems require the user to allow access to his or her social network profile information and at the same time associate that information with the user's identity. For instance, Facebook applications generally require the user to agree to give the application access to his/her information through Facebook's API, intrinsically tying such information to the user's identity. In the WhozThat and SocialAware systems, anyone near the mobile user can use a Bluetooth device to snoop a user's shared social network ID or eavesdrop on data sent openly over a wireless connection, since all data transmitted over the wireless connection is sent in the clear, although relatively weak provisions for link-layer encryption exist [16].

In a peer-to-peer context-aware mobile social network system such as SocialAware, we can track a user by logging the date and time that each mobile or stationary device detects the user's social network ID. By collecting such logs, we can construct a history of the locations that a user has visited and the times of each visit, compromising the user's privacy. Finally, given access to a user's social network ID, someone else could access that user's public information in a way that the user may not have intended by simply viewing that user's public profile on a social network Web site. We conclude that cleartext exchange of social networking IDs in systems such as WhozThat and SocialAware leads to unacceptable security and privacy risks, and allows the user's anonymity to be easily compromised. We call such problems that directly compromise a user's anonymity *direct anonymity attacks*.

Direct anonymity attacks are also possible in client-server mobile social network systems. While users' social network IDs are generally not directly exchanged between mobile devices in such systems, mobile or stationary devices can still track a user by logging the date and time that each device finds the user nearby. Since each device in these systems can find the social network user names and often full names of nearby users, the privacy of these users can be compromised. Thus, we have a direct anonymity issue - exposure of user names and locations in client-server systems allows the user's anonymity to be compromised.

B. The Indirect or K-Anonymity Problem

One worthwhile challenge is that of supporting complex mobile social networking applications with personal information without compromising the anonymity of the users providing the information. Even if the user does not directly provide his/her identification information, the user's provided social network information (such as preferences) may be mapped back to the user's identity through the social network site or information cached within mobile and stationary devices in the environment. The indirect anonymity problem exists when a piece of information indirectly compromises a user's identity. An example of this is when a piece of information unique to a user is given out, such as a list of the user's favorite movies, this information might then be easily mapped back to the user. The K-anonymity problem occurs when n pieces of information or n sets of related information can be used together to uniquely map back to a user's identity. Furthermore, if a set of information can only be mapped to a set of k or fewer sets of users, the user's anonymity is still compromised to a degree related to k. The challenge is to design an algorithm that can decide what information should and should not be given out in order to guarantee the anonymity of associated users. The abundance and diversity of social network information makes this privacy guarantee more complicated than it may initially appear. More formally, the particular problem is to find what personal information can be shared such that this information cannot be used to associate the user's identity with a specific context.

This problem is similar to previous K-anonymity problems related to the release of voter or hospital information to the public. However, it has been shown that by correlating a few data sets a high percentage of records can be "re-identified". A paper by Sweeney shows how this re-identification process is done using voter records and hospital records [17]. The K-anonymity problem in this paper is unique in that the standard K-anonymity guarantees that released information cannot distinguish between $k-1$ individuals associated with the released information. However, the problem discussed here does not involve the release of personal records but rather sets of aggregated information that may relate to sets of individuals that may or may not be associated with the released information. Therefore, the K-anonymity guarantee for our problem refers to the "*minimal*" number of indistinguishable unique sets that are sufficient to account for all released information. More precisely there must be no more than $k-1$ unique sets that are not subsets of each other and all other sufficient sets are supersets of some of the minimal sets.

Finding or defining this "*minimal*" set of sets is equivalent to the simplification of a Boolean algebra expression, in which the elements of all sufficient sets are connected by conjunction (AND) and all sets are logically disjunct (OR). The simplified form of this expression is defined as the "*minimal*" set of sets in which the simplified expression is made up of more than $k-1$ logically disjunct sets. A set of data for which more than $k-1$ *minimal* sets exist is admissible under a K-anonymity guarantee of k.

This problem can be phrased as an admissible set problem. Given two sets A and B where A is the set of all users and B is the set of all social network information that may be provided to a mobile social network application. The information in B has a many-to-many relation to A, since a user may have many pieces of information associated with him/her and many users may be associated with identical pieces of information. The problem is then to define an admissible set under a K-anonymity guarantee, which would define whether or not a subset x of B is admissible.

This paper presents this K-anonymity problem informally and proposes a solution that is currently being explored and implemented by the authors, however it does not formally solve this problem, which is proposed as an important open problem in the area of mobile social network privacy. We argue that this problem is important because it would provide an

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 3

alternative for users to take advantage of new mobile social network applications without compromising their privacy. The K-anonymity problem applies to both peer-to-peer and client-server mobile social network systems, since both systems involve sharing a user's social network profile data with other users of these systems.

C. Eavesdropping, Spoofing, Replay, and Wormhole Attacks

Once a user's social network ID has been intercepted in a peer-to-peer mobile social network system, it can be used to mount a replay and spoofing attack. In a spoofing attack, a malicious user can masquerade as the user whose ID was intercepted (the compromised user) by simply sending (replaying) the intercepted ID to mobile or stationary devices that request the user's social network ID. Thus, the replay attack, where the compromised user's ID is maliciously repeated, is used to perform the spoofing attack. Another specific type of replay attack is known as a wormhole attack [18], where wireless transmissions are captured on one end of the network and replayed on another end of the network. In a system such as WhozThat or SocialAware, a malicious user could use a wormhole attack to capture a user's ID and masquerade as that user in a different, perhaps distant, location. Since these systems are vulnerable to such replay and spoofing attacks, we can no longer trust that each user who participates in these systems is really who they claim to be. Therefore, the value of such systems is substantially diminished. Furthermore, these attacks could be used for a variety of nefarious purposes. For example, a malicious user could masquerade as the compromised user at a specific time and place while committing a crime. Clearly, spoofing attacks in mobile social networking systems present serious security risks.

In addition to intercepting a user's social network ID via eavesdropping of the wireless network, a malicious user could eavesdrop on information transmitted when a device requests a user's social network profile information from a social network server. For example, if a mobile device in a peer-to-peer system uses HTTP (RFC 2616) to connect to the Facebook API REST server [19] instead of HTTPS (RFC 2818), all user profile information requested from the Facebook API server is transmitted in cleartext and can be intercepted. Interception of such data allows a malicious user to circumvent Facebook's privacy controls, and access private user profile information that the user had no intention to share.

Eavesdropping, spoofing, replay, and wormhole attacks are generally not major threats to client-server mobile social network systems. These attacks can be defended against with the appropriate use of a robust security protocol such as HTTPS, in conjunction with client authentication using user names and passwords or client certificates. If a user's social network login credentials (user name and password, or certificate) have not been stolen by a malicious user and the user has chosen an appropriately strong password, then it is nearly impossible for the malicious user to masquerade as that user.

IV. Security and Privacy Solutions

We have designed and implemented a system, called the identity server, to address the security and privacy problems described previously. Our system assumes that each participating mobile device has reasonably reliable Internet access through a wireless wide area network (WWAN) cell data connection or through a WiFi connection. Mobile devices that lack such an Internet connection will not be able to participate in our system. Furthermore, we assume that each participating mobile device has a short-range wireless network interface, such as either Bluetooth or WiFi, for ad-hoc communication with nearby mobile and/or stationary devices. We describe the design and implementation of the identity server in this section.

A. Design of the Identity Server and Anonymous Identifier

As discussed in subsections III-A and III-C, the cleartext exchange of a user's social network ID presents significant privacy and security risks [20]. To address these risks, we propose the use of an *anonymous identifier*, or *AID*. The AID is a nonce that is generated by a trusted server, called the *identity server* (IS). Before a user's mobile device advertises the user's presence to other nearby mobile and stationary devices, it securely contacts the IS to obtain the AID. The IS generates a new AID for this mobile device using a cryptographic hash function such as SHA-1, with a random salt value. The IS associates the newly generated AID with the mobile device that requested the AID, and then returns the new AID to the mobile device. The user's mobile device then proceeds to share this AID with a nearby mobile and/or stationary device by launching a Bluetooth AID sharing service. After a nearby mobile or stationary device (device B) discovers this AID sharing service on the user's mobile device (device A), device B establishes a connection to the user's mobile device to obtain the shared AID. After the AID has been obtained by device B, device A requests another AID from the IS. This new AID will be shared with the next mobile or stationary device that connects to the AID sharing service on device A. While our design and implementation uses Bluetooth for AID sharing, we could also implement AID sharing using WiFi.

After the device B obtains the shared AID from device A, device B then proceeds to query the IS for the social network profile information for the user that is associated with this AID. Figure 1 shows the role of the IS in generating AIDs and processing requests for a user's social network information. Once the social network information for an AID has been retrieved by the IS, the IS removes this AID from the list of AIDs associated with the mobile user. Before the user's mobile device next advertises the user's presence using the Bluetooth AID sharing service, it will obtain a new AID from the IS as described above.

We permit multiple AIDs to be associated with a mobile user, which allows for multiple nearby mobile or stationary devices to obtain information about the user. To improve efficiency, the user's mobile device may submit one request for multiple AIDs to the IS, and then proceed to share each

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 4

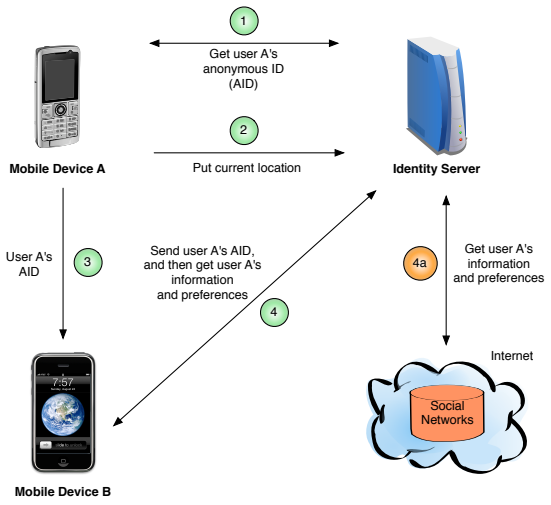

Fig. 1. Anonymous IDs and the Identity Server

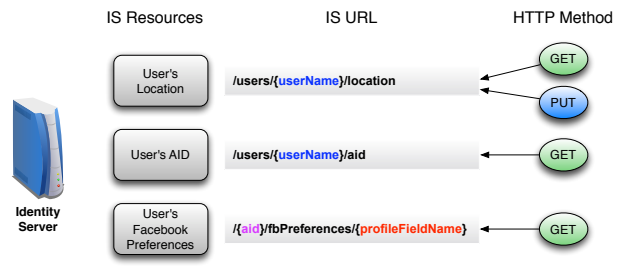

Fig. 2. Identity Server web-accessible resources

AID one at a time with other nearby devices. The IS sets a timeout value for each AID when the AID is created and provided to a user's mobile device. An AID times out if it is not "consumed" within the timeout period, that is, if the IS has not received a query for social network profile information for the user associated with this AID within the timeout period. Upon timeout of an AID, the IS removes the AID from the list of AIDs associated with the user. We use AID timeouts to prevent the list of AIDs associated a user from growing without bound.

The use of AIDs in our system provides important privacy features for mobile users. Since the mobile device shares only AIDs with other devices, a malicious user who has intercepted these AIDs cannot connect these AIDs to a particular user's social network identity. Furthermore, the IS does not support the retrieval of certain personally identifiable information from a user's social network profile, such as the user's full name, email address, phone number, etc. Since the IS does not support the retrieval of personally identifiable information, a device that retrieves social network information for the user associated with an AID is unable to connect the AID to the user's social network identity. Thus, only by compromising the IS can a malicious user tie an AID to a user's social network ID. We assume that the IS is a secure and trusted system, and that compromising such a system would prove to be a formidable task.

The use of IS and AIDs as we have described solves the direct anonymity problem. As the reader will see in subsection IV-C, the IS also addresses the indirect anonymity problem by providing a K-anonymity guarantee for information returned from users' social network profiles.

B. Implementation of the Identity Server

We have implemented the IS using the Java Standard Edition (SE) 5.0 platform. All IS services accessed by mobile and/or stationary devices are exposed as web services conforming to the REST architecture [21]. We used the open source Reslet framework [22] for Java to develop the IS. We expose each resource on the IS, including a mobile user's AID, a mobile user's current location, and the Facebook profile information for a mobile user, as separate URL-accessible resources supporting HTTP GET, POST, and PUT methods as appropriate. Figure 2 shows the web-accessible resources exposed on the IS, along with the HTTP methods supported by each resource. The body of each HTTP request is encoded using JSON (RFC 4627). All web service network traffic between the IS and other mobile/stationary devices is encrypted using HTTPS, and access to all resources is authenticated using HTTP basic access authentication (RFC 2617).

Each mobile user must sign up for a user account on the IS prior to participation in our system. During the signup process, the user provides his/her Facebook user ID (we can obtain this using Facebook Connect [23]), and chooses a user name and password. The user's user name and password are securely stored on the user's mobile device, and are used to authenticate with the IS and obtain access to the guarded web resources on the IS for the device's current location, the user's AID, and the user's Facebook profile information. Access to the web resources for the user's AID and current location is available only to the user herself/himself, and no other entity save for the logic implemented on the IS. Access to the web resource for the user's Facebook profile information (we call this user "user A") is provided to any authenticated user with a user account on the IS, provided that the authenticated user's device is within an acceptable range of user A's mobile device. See below for more information on location-based access control for a user's Facebook profile.

We implement all data persistence on the IS using the open source SimpleJPA tool [24]. SimpleJPA is a Java Persistence API (JPA) [25] implementation for Amazon's SimpleDB [26]. By using SimpleDB, we take advantage of Amazon's simple, scalable, and reliable distributed database system. SimpleDB structures all data into domains. Our use of SimpleJPA and SimpleDB allows us to easily launch new IS instances that all communicate with the same set of domains backed by a shared distributed database, providing for an implementation of our system that is quite scalable.

AIDs for each mobile user are generated on the IS using

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 5

TABLE II
K-ANONYMITY EXAMPLE DATA SET

Name	Color	Letter	Number
Bill	Red	A	1
Fred	Green	A	2
Jon	Green	B	2
Joe	Red	C	1

the SHA-1 cryptographic hash function with a 16-byte random salt value. A new AID for a user is generated on the IS each time that the user's mobile device requests an AID. The IS maintains a mapping of AIDs to users' Facebook IDs in the persistence layer. As mentioned previously, multiple AIDs can be associated with a single mobile user, and each AID is assigned a timeout value by the IS. In our implementation, we set the AID timeout value to 30 seconds. The Facebook REST API web service [19] is used by the IS to obtain the content of fields of a user's Facebook profile. Each time that a mobile or stationary device (device B) requests the Facebook preferences for a mobile user (using device A), the IS checks the locations of devices A and B to verify that the these devices are within an acceptable range of each other before returning the requested information. In our IS implementation, we set this maximum acceptable range to 20 meters.

C. K-Anonymity

We begin our discussion of a solution to the K-anonymity problem with the following example. Consider the example data set in table II. If the set $(Red, A, 1)$ is released or given to a third-party application it can be related back to the *minimal* unique sets $(Bill)$ and $(Fred, Joe)$ implying that at least, Bill **OR** Fred **AND** Joe are associated with the data. This does not rule out the possibility of other super-sets that include these minimal sets such as $(Bill, Fred)$ or $(Bill, Fred, Joe)$, however it implies that one of two minimal sets must be associated with the data. This would be an example of K-anonymity where $k \leq 2$, such that *more than* $k-1$ *minimal* sets are indistinguishable.

Obviously if only two sets of users map to a piece of data, one other piece of data within the provided set or any subsequent set, which also contains the same piece of data, may be used to distinguish which user to associate with the data. Therefore, algorithms to determine admissible sets should maintain state between a number of n sequential sets $x_1, x_2, ...x_n$ of information, guaranteeing that at least k minimal sets of users are always indistinguishable as related to the n sequential sets of related data $x_1, x_2, ...x_n$. We are exploring the use of logic simplification algorithms such as Quine-McCluskey [27] to solve this problem quickly.

In order to use existing logic simplification software, the relationship between users and their preferences must be modeled as a set of truth cases. This can be done many ways. One example would be to model the preference-user couples as nodes in a graph. First, the nodes are partially-ordered by preference and then each node is connected to all nodes preceding and following it. The set of all truth cases would be the superset of all paths from beginning nodes (those with the first ordered preference) to the end nodes (those with the last ordered preference). Each path would map to a conjunctive clause of literals (one literal per node) in the final disjunctive normal form (DNF) Boolean expression. All paths/clauses would then be disjunctive causing the overall expression to be true if any of the truth cases resolved to true.

We have implemented this basic approach to verifying K-anonymity guarantees and are beginning real-time performance tests using the Quine-McCluskey algorithm [27] for logic simplification. We are using a basic context-aware multimedia application that samples the media (music and movie) preferences of users within a local area. All user queries go through the IS, which implements the K-anonymity guarantee. Initial tests have shown the solution is feasible for user groups as large as most social network friend lists (consisting of 200–300 friends), which have made K-anonymity guarantees possible with $k = 20$.

D. Eavesdropping, Spoofing, Replay, and Wormhole Attacks

Our security and privacy solutions provide several security features that address the security threats outlined in subsection III-C. The use of AIDs prevents spoofing and replay attacks. Since AIDs, instead of social network IDs (such as Facebook IDs), are shared by the mobile device, a malicious user cannot spoof the social network identity of another user. By using a cryptographic hash function with a random salt value to generate AIDs for each mobile user, and continuously generating new AIDs upon request as AIDs timeout or are consumed by other devices, we prevent replay attacks whereby a malicious user may attempt to capture and reuse a sequence of AID values previously shared by a mobile device.

Before a mobile device advertises its AID, it must inform the IS of its current location. We assume that all mobile devices that participate in our system use a secure positioning system, such as [28]. Such a system prevents location spoofing. Our assumption of a secure positioning system also prevents wormhole attacks [18] using AIDs, whereby a malicious user captures AIDs shared by a mobile device in one location and retransmits them to a device in another location. A malicious user cannot capture an AID shared by a mobile device and retransmit it somewhere else for sharing with a distant device (called device B), since the IS verifies that the mobile device corresponding to this AID is within an acceptable range of device B whenever device B attempts to obtain social network information for the mobile user associated with this AID.

We provide reasonable protection against eavesdropping in our system by encrypting all network traffic between mobile/stationary devices and the IS using a technology such as HTTPS. Our use of encryption prevents interception of user credentials and all other information passed between these components, including a user's private social network profile information.

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 6

E. Trust Networks and Onion Routing

One way to support privacy in social network applications is to transfer information using a trusted peer-to-peer network [29]. Such a network would require a trust network much like that used by Katz and Goldbeck [30] in which social networks provided trust for default actions on the web. Moreover, in a mobile social network application, nodes could not only share their information directly but could give permission to their trusted network to share their information. This approach was used in the OneSwarm [31] system to allow peer-to-peer file sharing with privacy settings that allowed the user to share data publicly, just with friends, or even with a chosen subset of those friends. However, such a model has obvious problems if any nodes are compromised since information is easily associated with its source.

These peer-to-peer networks could be made anonymous through the use of onion routing [32]. The Tor network [33] uses onion routing to allow nodes to send data anonymously. Through the use of layers of encryption that are decrypted at selected routers along a virtual route, routing nodes cannot directly relate the information at the destination to its source. If data was shared in this manner it would not be so easy to identify the source of the information, protecting the direct anonymity of the user. We are currently exploring the use of trust networks and onion routing in terms of taking a more decentralized approach to protecting user anonymity that does not require trust of the social network (such as Facebook) itself [29].

V. Conclusion

We have identified several important privacy and security issues associated with LAMSN systems, along with our work on novel solutions for these issues. Our solutions support anonymous exchange of social network information with real-world location-based systems, enabling context-aware systems that do not compromise users' security and privacy. We hope that our work will convince users and developers that it is possible to move forward with creative mobile social network applications without further compromising user security and privacy.

References

[1] A. Beach, M. Gartrell, S. Akkala, J. Elston, J. Kelley, K. Nishimoto, B. Ray, S. Razgulin, K. Sundaresan, B. Surendar, M. Terada, and R. Han, "Whozthat? evolving an ecosystem for context-aware mobile social networks," *IEEE Network*, vol. 22, no. 4, pp. 50–55, July-August 2008.

[2] N. Eagle and A. Pentland, "Social serendipity: Mobilizing social software," *IEEE Pervasive Computing*, vol. 4, no. 2, April-June 2005.

[3] "Mobile browsing by platform market share," http://marketshare.hitslink.com/mobile-phones.aspx?qprid=55&sample=31.

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

[5] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proceedings of the 5th ACM/USENIX Internet Measurement Conference (IMC'07)*, October 2007.

[6] "Global internet use reaches 1 billion," http://www.comscore.com/press/release.asp?press=2698.

[7] C. M. Gartrell, "Socialaware: Context-aware multimedia presentation via mobile social networks," Master's thesis, University of Colorado at Boulder, December 2008.

[8] "Blackberry facebook application," http://www.facebook.com/apps/application.php?id=2254487659.

[9] E. Miluzzo, N. D. Lane, S. B. Eisenman, and A. T. Campbell, "Cenceme - injecting sensing presence into social networking applications," in *Proceedings of the 2nd European Conference on Smart Sensing and Context (EuroSSC 2007)*, October 2007.

[10] "Brightkite," http://brightkite.com.

[11] "Loopt," http://www.loopt.com.

[12] A. Tootoonchian, K. K. Gollu, S. Saroiu, Y. Ganjali, and A. Wolman, "Lockr: social access control for web 2.0," in *WOSP '08: Proceedings of the first workshop on Online social networks*. New York, NY, USA: ACM, 2008, pp. 43–48.

[13] S. Ahern, D. Eckles, N. S. Good, S. King, M. Naaman, and R. Nair, "Over-exposed?: privacy patterns and considerations in online and mobile photo sharing," in *CHI '07: Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA: ACM, 2007, pp. 357–366.

[14] L. P. Cox, A. Dalton, and V. Marupadi, "Smokescreen: flexible privacy controls for presence-sharing," in *MobiSys '07: Proceedings of the 5th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2007, pp. 233–245.

[15] J. Manweiler, R. Scudellari, Z. Cancio, and L. P. Cox, "We saw each other on the subway: secure, anonymous proximity-based missed connections," in *HotMobile '09: Proceedings of the 10th workshop on Mobile Computing Systems and Applications*. New York, NY, USA: ACM, 2009, pp. 1–6.

[16] A. Becker, "Bluetooth security & hacks," http://gsyc.es/~anto/ubicuos2/bluetooth_security_and_hacks.pdf.

[17] L. Sweeney, "Uniqueness of simple demographics in the U.S. population," in *LIDAPWP4*, 2000.

[18] R. Maheshwari, J. Gao, and S. Das, "Detecting wormhole attacks in wireless networks using connectivity information," in *26th IEEE Conference on Computer Communications (INFOCOM 2007)*, May 2007.

[19] "Api - facebook developers wiki," http://wiki.developers.facebook.com/index.php/API.

[20] S. Guha, K. Tang, and P. Francis, "Noyb: privacy in online social networks," in *WOSn '08: Proceedings of the first workshop on Online social networks*. New York, NY, USA: ACM, 2008, pp. 49–54.

[21] R. Fielding, "Representational state transfer (rest)," http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.

[22] "Restlet - lightweight rest framework," http://www.restlet.org.

[23] "Facebook connect," http://developers.facebook.com/connect.php.

[24] "Simplejpa - java persistence api for amazon simpledb," http://code.google.com/p/simplejpa/.

[25] "Java persistence api," http://java.sun.com/javaee/technologies/persistence.jsp.

[26] "Amazon simpledb," http://aws.amazon.com/simpledb/.

[27] "Quine-mccluskey algorithm (java)," http://en.literateprograms.org/Quine-McCluskey_algorithm_(Java).

[28] S. Čapkun, K. Rasmussen, M. Čagalj, and M. Srivastava, "Secure location verification with hidden and mobile base stations," *IEEE Transactions on Mobile Computing*, vol. 7, no. 4, pp. 470–483, April 2008.

[29] L. Cutillo, R. Molva, and T. Strufe, "Privacy preserving social networking through decentralization," in *WONS 2009, 6th International Conference on Wireless On-demand Network Systems and Services*. New York, NY, USA: ACM, 2007, pp. 357–366.

[30] Y. Katz and J. Golbec, "Using social network-based trust for default reasoning on the web," *Journal of Web Semantics*, 2007.

[31] T. Isdal, M. Piatek, A. Krishnamurthy, and T. Anderson, "Friend-to-friend data sharing with OneSwarm," Department of Computer Science, University of Washington, Tech. Rep. Technical report, UW-CSE, February 2009.

[32] M. G. Reed and P. F. Syverson, "Onion routing," in *AIPA*, 1999.

[33] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: the second-generation onion router," in *SSYM'04: Proceedings of the 13th conference on USENIX Security Symposium*. Berkeley, CA, USA: USENIX Association, 2004, pp. 303–320.

Authorized licensed use limited to: Callie Pendergrass. Downloaded on November 17,2023 at 00:11:58 UTC from IEEE Xplore. Restrictions apply.

IPR2024-00399
Apple EX1040 Page 7