University of Colorado, Boulder
CU Scholar

Computer Science Technical Reports | Computer Science

Summer 6-1-2009

Secure SocialAware: A Security Framework for Mobile Social Networking Applications ; CU-CS-1054-09

Aaron Beach
University of Colorado Boulder

Mike Gartrell
University of Colorado Boulder

Baishakhi Ray
University of Colorado Boulder

Richard Han
University of Colorado Boulder

Follow this and additional works at: http://scholar.colorado.edu/csci_techreports

Recommended Citation
Beach, Aaron; Gartrell, Mike; Ray, Baishakhi; and Han, Richard, "Secure SocialAware: A Security Framework for Mobile Social Networking Applications ; CU-CS-1054-09" (2009). *Computer Science Technical Reports.* Paper 983.
http://scholar.colorado.edu/csci_techreports/983

This Technical Report is brought to you for free and open access by the Computer Science at CU Scholar. It has been accepted for inclusion in Computer Science Technical Reports by an authorized administrator of CU Scholar. For more information, please contact jennifer.chan@colorado.edu.

Secure SocialAware: A Security Framework for Mobile Social Networking Applications

Aaron Beach, Mike Gartrell, Baishakhi Ray, Richard Han
Contact: Aaron.Beach@colorado.edu, Mike.Gartrell@colorado.edu

Department of Computer Science
University of Colorado at Boulder

Technical Report CU-CS-1054-09

June 2009

Secure SocialAware: A Security Framework for Mobile Social Networking Applications

Aaron Beach, Mike Gartrell, Baishakhi Ray, Richard Han
Department of Computer Science, University of Colorado at Boulder
Contact: Aaron.Beach@colorado.edu, Mike.Gartrell@colorado.edu

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 position and preferences. However, current models for exchange of this information require users to compromise their privacy. We present a framework called Secure SocialAware (SSA) which allows for the interaction of social network information with real-world location-based services without compromising user privacy and security. Through exchanging an encrypted nonce (EID) associated with a verified user location, SSA allows location-based services to query the local area for social network information without disclosing user identity or any set of information which could be positively matched to users. We argue that it is important that such a framework be accepted as mobile social networks are growing exponentially.

Keywords

design, security, privacy, mobile computing, social networks

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks: General**]: security and privacy.

1 Introduction

Online social networks provide the opportunity to substantially enrich interaction with ubiquitous computing environments by integrating social context information into local interactions. New middleware infrastructure must be introduced to harness the rich personal preference and friendship information contained in social networking sites so that interaction with the local ubiquitous computing environment is conveniently personalized. Location-aware mobile social networks 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.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

This paper describes Secure SocialAware (SSA), a general framework to support secure and anonymous exchange of social network or other personal information throughout a user's physical location. A system may use such an architecture to gather data on the history and preferences of users in a particular area without compromising those users' privacy. For instance a media player may query the media preferences of users within a certain geographical area (as large as a city or as small as a room), and then use that information to automatically distribute media in real time. The system may process the media preferences in different ways to achieve different goals, however it should not be able to use this information to identify the users in the specified area. The abundance of such Internet-accessible information and the abundance of diversity among this information make this privacy guarantee more complicated than it may sound at first. SSA seeks to provide location-based personal information which may be used to discern the possibility of a specific user's presence, without linking that presence with a specific user's identity. That is, the information provided by SSA may used to discern if a user *is not* in the area, but never whether the user *is* in the area.

Privacy or privacy threats are often referred to in the discussion of social networks and mobile computing, however the examples given are often spurious or incomplete. Examples themselves are rarely enough to fully describe a range of privacy threats within an area as wide as mobile computing, however they are often the quickest and sometimes only way to convey a sense of the existing problems within the dynamic space of social networks and mobile computing. Initially the authors of this paper sought to better formalize the threats which motivated this work, however the threats seem as dynamic and forthcoming as the technology and usage habits associated with them. As such, examples are what the authors have to work with and a few of these will be discussed below to give the reader a sense of the novelty and urgency motivating this work.

Given the identity of a social network user, be it a name or worse, an identity number, certain “public” information can immediately be accessed, along with more personal information depending on source-specific settings. This information has been shown to be easily used to track or steal a user’s identity. Often users of social networks and mobile apps do not understand the details of how their information is accessed. For instance, Facebook users may assume that “friending” another user gives that user special access to their data. Indeed they would be right, however they are not just giving that user special access to the data, they are giving that user the “right” or at least ability to use the data and give it away to any third-party applications they choose to use. Currently, third-party application platforms on social networks are designed to exchange services (the application services) for access to users’ data. This model is now being extended to mobile-phone applications, such as those that Facebook claims are being developed by over 150 mobile operators in over 50 countries [3]). While many Internet users have now become more wary of what websites they give their information to, they are often not very careful about what third-party applications they install on their phones or add to their social network profile. The Facebook Statistics page [3] states that there are over 660,000 application developers for their site and that more than 70% of Facebook users interact with the applications created by these developers. Likewise the iPhone is seeing unexpectedly high usage of mobile apps, with over 1 billion downloaded from the App Store [4]. Obviously users are showing interest in third-party applications which use their social networking information or interact with their mobile devices. SSA allows this interaction without sacrificing the user’s identity or compromising access to private data which users prefer not to share.

This work draws on some previous privacy research in both Location Based Services and Social Networks. Previous work at Duke University [5] [6] 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. These works do 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 [5] presents a protocol by which devices may broadcast identifiers which 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. SSA, however, differs in that it seeks to hide the user’s identity while distributing certain personal information. The Serendipity project [2] at MIT is an example of a mobile social network in which users used their phones to send and receive social context information, and detect nearby users. Commercial examples of mobile social networks include Loopt [7] and Brightkite [8]. Indeed, many new applications allow users to use existing personal information in much the same way as these mobile social networks.

2 Location-Aware Mobile Social Network Systems

This section intends to give the user an introduction to the type of systems which SSA supports. Generally these systems allow personal information (usually from social networks) to be used by third-party applications. First we will discuss how third-party applications already function within social networks and on smartphones. Next, a few prototype architectures will be discussed along with the security and privacy problems associated with each architecture.

2.1 Online Social Network Applications

Currently social network sites such as Facebook [9] and Myspace [10] allow users to add applications or widgets which extend how users interact with the system and its data. For instance on Facebook you can download applications which search out which of your friends are also friends with each other and then proceed to automatically generate clustered friend graphs. These applications require (and are given access) not only to the application user’s information, but to all information which that user has access to, such as if the user’s friends are friends of each other.

2.2 Mobile Phone Applications

The iPhone App Store sells applications which can both access existing social networks (Facebook’s “iPhone App”), access mobile social networks (Loopt), and allow users to find and contact users around them (WhosHere App [11]). These applications use access to the Internet and the iPhone’s location service, which takes advantage of GPS or triangulation using cell towers or WiFi hotspots. Tying location services together with social network information allows applications in the real world to react to their local environment in user-specific ways.

2.3 Peer-to-Peer Mobile Social Networking: WhozThat

In [1] a vision was presented for WhozThat, a system for answering the basic social question of “Who’s that?”. WhozThat ties together existing online social networks with mobile smartphones, and provides an infrastructure for building a new class of context-aware applications that leverage social network information. Social networks provide access to an extensive collection of personal information about users, including name, gender, contact information, interests, music preferences, movie preferences, book preferences, and friendship connections with others, which represent but a few of the fields that users populate on their social network profiles.

WhozThat uses a peer-to-peer model for sharing users’ social network identifiers among smartphones and other devices, where social network IDs are broadcast using a wireless technology such as Bluetooth or WiFi. WhozThat’s identity sharing protocol enables applications such as chat, social network profile viewing, and user search based on specified criteria. The peer-to-peer exchange of information in this system is shown in figure 1. With WhozThat, a user can walk into a shared physical space and determine who in the room has interests similar to the user’s. The user can then proceed to view the social network profile of that person and even contact him/her directly through chat.

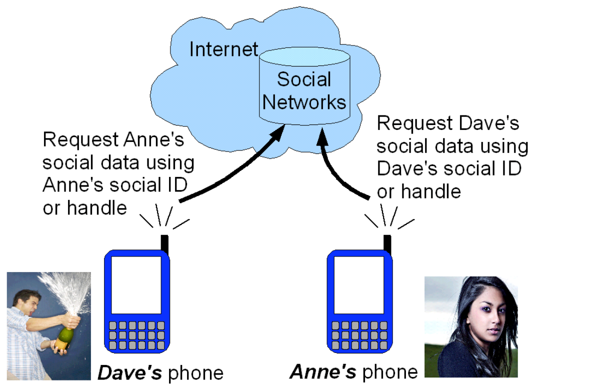

Internet
Social Networks
Request Anne's social data using Anne's social ID or handle
Request Dave's social data using Dave's social ID or handle
Dave's phone
Anne's phone

Figure 1. Peer-to-peer information exchange in WhozThat

The WhozThat prototype system was implemented by advertising a plain-text representation of the user's Facebook ID over Bluetooth. When a user enters a new space, his/her mobile device queries the list of Bluetooth devices in the room and then queries each device for its user's Facebook ID. This ID is then used to access each user's Facebook profile information using the Facebook API web service [12] through an Internet connection. This gives users access to whatever information they could normally access if using Facebook, without breaking Facebook privacy rules. However, simply not breaking Facebook privacy rules still allows the user's own privacy to be compromised. Given a user's Facebook ID one could easily access the user's public information and approximate the location of that user. This kind of system fits the existing Facebook application model while exhibiting severe and obvious privacy problems.

2.4 Context-Aware Services: Basic SocialAware Architecture

A framework for building context-aware mobile social networking services in a ubiquitous computing environment, called SocialAware, is described in [13]. SocialAware applications use WhozThat's identity sharing protocol for exchanging users' social network IDs among mobile and stationary devices. SocialAware differs from WhozThat in that it is built around a stationary system (called the Stationary Component or SC) for which each mobile client (called the Mobile Component or MC) is an information provider to the stationary system. The information provided by each Mobile Component is the user's social network ID and the information being used by the Stationary Component is the user's social network profile information associated with his/her social network ID. Two prototype SocialAware applications are described in [13], both of which present multimedia (either movie trailers or music) in the vicinity of the mobile clients that fits the media preferences of each mobile client user as expressed on their Facebook profiles. In order to determine what actions to take based on the user's preferences, the SC consults one or more domain-specific web service recommender systems. One example of such a recommender systems is the Netflix REST API web service [14].

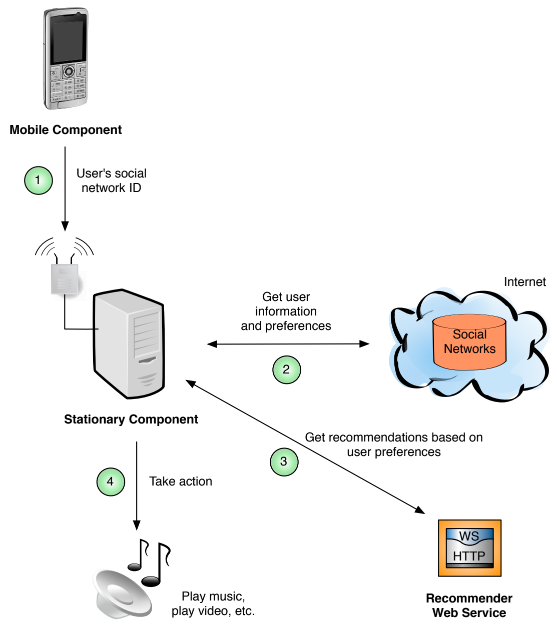


Figure 2. Basic SocialAware architecture

Figure 2 shows the SocialAware system model. This system can be seen as a simple model for the basic components of any context-aware system that leverages social networks, in which the context is the user's location and personal information which must be exchanged with a central system.

2.5 Privacy & Security Issues

The information exchange model of the systems discussed in this section provide little protection for user's privacy. These systems require the user to allow access to their 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, anybody 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 without encryption.

Once a user's social network ID has been intercepted, 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 systems 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 [15], 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 the systems described in this section 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 a 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 a 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 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.

While the existing context-aware mobile social networking systems described in this section provide examples of how social network information can be used to increase the value of users' physical or Internet experiences by tailoring these experiences to users' specific preferences, they also demonstrate the dangers posed by such systems when implemented in the most naive or direct way. These dangers are the primary motivation for SSA. Namely, we seek to provide personally tailored context-aware services to users without compromising their privacy.

3 Secure SocialAware Architecture

The Secure SocialAware (SSA) framework consists of three major components: the stationary component (SC), the mobile component (MC), and the authentication server (AS). The SC is embedded into the user's environment, while the MC component resides on mobile devices. The SC is responsible for detecting the presence of users, obtaining information about these users from the AS, and performing actions based on this information. The MC on a user's mobile device is responsible for advertising that user's encrypted identifier (EID) using a wireless technology such as Bluetooth so that it is available to the SC. The AS generates unique EIDs for each MC, and provides a service to the SC so that the SC can obtain the user's information from a social networking web site given an EID.

The AS allows SCs and MCs to join the SSA system by signing up for stationary and mobile user accounts, respectively. After a mobile user signs up for a user account, the AS assigns that user a unique EID. The AS provides secure services that the MC uses to determine its EID and inform the AS of its current location. The AS also provides a secure service that the SC uses to retrieve the mobile user's information from a social networking Web site given that user's EID. To obtain user information from a social networking

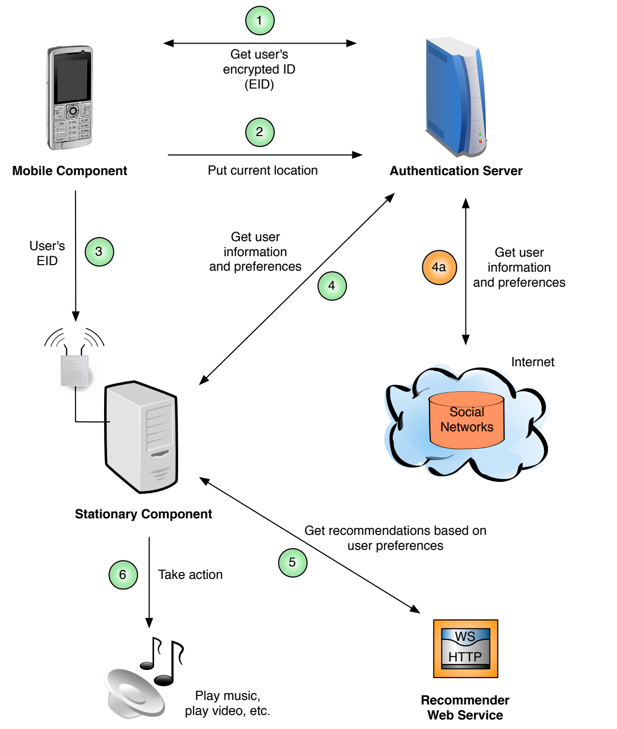

1
Get user's encrypted ID (EID)
2
Put current location
Mobile Component
Authentication Server
User's EID
3
Get user information and preferences
4
4a
Get user information and preferences
Internet
Social Networks
Stationary Component
Get recommendations based on user preferences
5
6
Take action
WS
HTTP
Play music, play video, etc.
Recommender Web Service

Figure 3. Secure SocialAware architecture

site, the AS maintains a mapping of EIDs to social network profile identifiers. Figure 3 shows the architecture of SSA.

3.1 Stationary Component

The SSA framework provides a stationary component that is embedded into the user's environment. In the SocialAwareFlicks example application discussed in the *Implementation and Evaluation* section of this paper, this component detects the EID of each nearby user and obtains information about the users' movie preferences from their Facebook profiles by accessing a secure service provided by the AS.

After obtaining information about detected users from a social networking Web site, the SC must use this information to infer something about the users and take appropriate action based on this inference. For the SocialAwareFlicks application, an external Web service is accessed to obtain movie recommendations based on the preference information provided on a user's Facebook profile. After obtaining these recommendations for each user, we use a set of heuristics to create a movie trailer playlist that accommodates the possibly varying tastes of the detected users in the group. The *Implementation and Evaluation* section describes the implementation of the SC for the SocialAwareFlicks application in more detail.

The SC is assumed to be a trusted component in SSA. As such, each SC that joins the SSA system must sign up for a SC user account on the AS. During the signup process, the SC administrator selects a user name and password, which together form the SC user credentials. These user credentials are used to authenticate access to all AS services used by the SC. All communication between the AS and SC occurs over an encrypted connection, using a technology such as HTTPS. In addition to selecting a user name and password, the SC administrator also sets the location coordinates for the SC during the signup process. Since the SC is a trusted component, we assume that the SC location is not falsified.

3.2 Mobile Component

The mobile component of the SSA framework is found on a user's mobile device. The MC shares the mobile user's EID and the current timestamp with the SC. This information will typically be shared with the SC using a short-range wireless technology such as Bluetooth, although other mechanisms may be used since SSA treats the ID sharing technology as an abstraction. For instance, while we are using a wireless peer-to-peer model to exchange EIDs, SSA can support systems which exchange EIDs through a secure server based system.

The MC must join the SSA system in order to obtain EIDs from the AS. The mobile user joins the SSA system by signing up for a MC user account on the AS. During the signup process, the mobile user selects a user name and password, which form the MC user credentials. After account signup has been completed, these user credentials are used to authenticate access to all AS services used by the MC. All communication between the AS and SC occurs over an encrypted connection, using a technology such as HTTPS. Before the MC shares an EID, it sends a service request to the AS to obtain the current EID for this mobile user. The generation of this EID on the AS is explained in the *Authentication Server* subsection of this paper, found below. The MC also uses a secure service on the AS to inform the AS of its current location.

The SSA framework also allows the user to interact with the local context-aware service using the MC. In SocialAwareFlicks, the user might use the MC to view information about the movie trailer that is currently playing, such as movie title and release date, and request that SocialAwareFlicks skip the remainder of the current movie trailer and begin playing the next movie trailer in the playlist. SSA applications treat the mechanism used for interaction with the user as an abstracted bidirectional communication channel. This abstraction is flexible enough to support user interaction over a variety of network technologies, including short-range wireless technologies such as Bluetooth or WiFi, or wireless wide area network (WWAN) connectivity available on many smartphones.

The broadcasting of an EID gives the MC *active* control over its interaction with the SSA system. Rather than setting a flag or updating a server resource which passively accepts requests for social network information, the AS requires that a nonce (EID) is actively sent from the MC. This makes it impossible for the MC to be locked in an accepting state or be locked in a specific location. A passive system may result in such a locked state if the MC loses communication with the AS through a lost signal or loss of power. A malicious user may intentionally attack a passive system through blocking MC communication. The active EID broadcast protects against such an attack.

3.3 Authentication Server

The AS provides for nearly all of the security features found in SSA. The AS is a web-accessible server that provides services for both the SC and the MC. Upon MC user account signup, the AS assigns an initial EID to the mobile user. The AS also stores the current location of a particular MC, as specified by the MC, and the user name and password specified by the mobile user during account signup. Upon SC user account signup, the AS stores the user name, password, and SC's location as specified by the SC administrator during account signup. The AS is assumed to be a trusted component in SSA.

Before the MC can share it's EID with a nearby SC, the MC must retrieve it's current EID from the AS and inform the AS of it's current location using secure services provided by the AS. Access to these secure services is authenticated using the MC's user credentials, and all network traffic for AS services is encrypted as mentioned above. The EID for an MC user is updated to a new unique value, using a cryptographic hash function with a random salt value, whenever the SC accesses the AS service which provides a user's social network profile information. This EID update-logic prevents replay attacks, since a stream of previously shared EIDs can never be reused.

The AS provides a secure service to the SC that allows the SC to retrieve social network profile information for an MC user using that user's EID. Access to this service is authenticated using the SC account's user credentials, and all network traffic for this service is encrypted. The AS computes the distance between the SC and the specified MC user, and will only return the requested MC user's social network profile information if the mobile device is within some specified range of the SC. If the mobile device is outside of this maximum acceptable range, then the request will be denied. In our implementation of SSA, we define this maximum acceptable range to be 20 meters.

Figure 4 shows the architecture of the AS. In this figure, the AS will perform either step 3b or step 3c after the SC requests user information and preferences (step 3), depending on whether or not the MC is located within the maximum acceptable range of the SC as determined in step 3a.

3.4 Security and Privacy Features

The SSA provides several security features that address the security threats outlined in the subsection of this paper titled *Privacy & Security Issues*. The use of EIDs prevents spoofing and replay attacks. Since EIDs, instead of social network IDs (such as Facebook IDs), are shared by the MC, 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 each mobile user's EID, and continuously updating this EID with each use of an EID value by the SC, we prevent replay attacks whereby a malicious user may attempt to capture and reuse a sequence of EID values previously shared by an MC.

We assume that all mobile devices that participate in SSA use a secure positioning system, such as [16]. Such a system prevents location spoofing. Our assumption of a secure positioning system also prevents wormhole attacks [15] using EIDs, whereby a malicious user captures EIDs shared by a

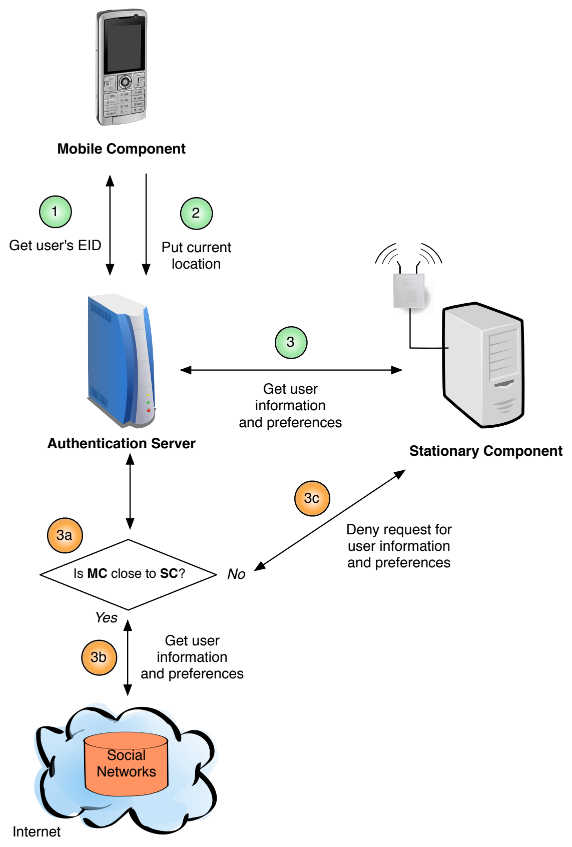


Figure 4. SSA Authentication Server architecture

MC in one location and retransmits them to an SC in another location. A malicious user cannot capture an EID shared by an MC and retransmit it somewhere else for sharing with a distant SC, since the AS verifies that the mobile device corresponding to this EID is within an acceptable range of the SC whenever the SC attempts to obtain social network information for the mobile user.

We provide reasonable protection against eavesdropping in SSA by encrypting all network traffic between the MC and the AS, and between the SC and the AS, using a technology such as HTTPS. Our use of encryption prevents interception of user credentials and all other information passed between these components.

The use of EIDs in our system provides important privacy features for SSA mobile users. Since the MC shares only EIDs with the SC, a malicious user who has intercepted these EIDs cannot connect these EIDs to a particular user's social network identity. Furthermore, the AS 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 AS does not support the retrieval of personally identifiable information, the SC is unable to connect an EID to a user's social network identity. Thus, only by compromising the AS can a malicious user tie an EID to a user's social network ID. Again, we assume that the AS is a secure and trusted system, and that compromising such a system would prove to be a formidable task.

4 Implementation and Evaluation

This section describes the implementation of the SSA framework, an example prototype application called SocialAwareFlicks that uses SSA and shows the feasibility of the framework, and an evaluation of our SSA implementation.

4.1 Implementation of the SSA Framework

The AS and SC were implemented using the Java Standard Edition (SE) 5.0 platform, while the MC was implemented using the Java Micro Edition (ME) Mobile Information Device Profile (MIDP) 2.0 platform. All interactions between the SC and MC are managed using Bluetooth. The BlueCove JSR-82 Emulator [17] tool was used to emulate all Bluetooth connectivity.

The Marge framework [18] was used to implement all of the Bluetooth communication logic in SSA. The Java API for Bluetooth (JSR 82) [19] is fairly low-level and reasonably difficult to use for those who are not familiar with the technical details of the Bluetooth standard. Marge provides an abstraction layer above JSR-82 that hides the low-level details of managing Bluetooth connections, message exchanges, protocols, device discovery, and service discovery.

After the MC user signs up for a mobile user account on the AS, the MC updates its location by accessing the appropriate web service on the AS, retrieves its EID from the AS by using a web service, and then proceeds to launch a SSA Bluetooth service that enables the user's EID to be shared with the SC. The MC uses the Marge framework to set up and launch the SSA Bluetooth service.

The SC detects the EIDs shared by nearby MCs by searching for available mobile devices running the SSA Bluetooth service. Since Bluetooth radios on mobile phones are Bluetooth Class 2 devices with an approximate range of 10 meters, only nearby users will be detected. The SC registers several listener objects with the Marge framework to manage the search for available mobile devices and to manage the state of the connection to the mobile device. After detecting an appropriate mobile device, we query the device to determine the user's EID. In the current implementation, the user's EID is encoded as the Bluetooth service name of the SSA Bluetooth service running on the mobile device. We are able to identify the SSA Bluetooth service by looking for a particular universally unique identifier (UUID) advertised by all mobile devices that run this service.

All AS services accessed by the MC and SC are exposed as web services conforming to the REST architecture [20]. We used the open source Reslet framework [21] for Java to develop the AS. We expose each resource on the AS, including a MC user's EID, a MC user's current location, and the Facebook profile information for a MC user, as separate URL-accessible resources supporting HTTP GET, POST, and PUT methods as appropriate. Figure 5 shows the web-accessible resources exposed on the AS, along with the HTTP methods supported by each resource. The body of

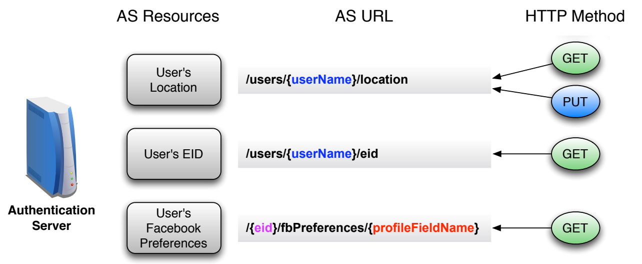


Figure 5. Authentication Server web-accessible resources

each HTTP request is encoded using JSON [22]. All web service network traffic between the AS and SC, and between the AS and MC, is encrypted using HTTPS, and access to all resources is authenticated using HTTP basic access authentication [23].

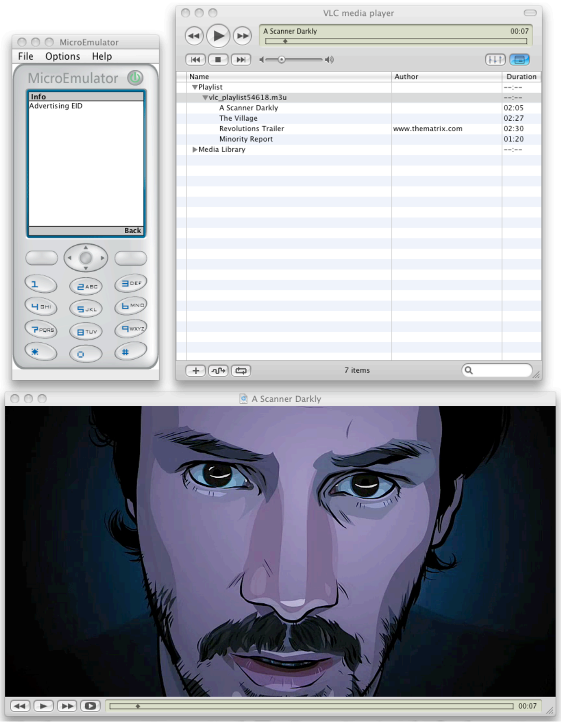
We implement all data persistence on the AS using the open source SimpleJPA tool [24]. SimpleJPA is a Java Persistence API (JPA) [25] implementation for Amazon's SimpleDB [26]. By using SimpleDB, we take advantage of Amazon's simple, scalable, and reliable distributed database system. SimpleDB structures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

EIDs for each MC are generated on the AS using the SHA-1 cryptographic hash function with a 16-byte random salt value. A new EID for a user is generated on the AS each time that the SC requests the Facebook profile information for a user. The AS maintains a mapping of EIDs to Facebook IDs in the persistence layer. The Facebook REST API web service [12] is used by the AS to obtain the content of fields on a user's Facebook profile. Each time that a SC requests the Facebook preferences for an MC user, the AS checks the locations of the MC and the SC to verify that the MC is within an acceptable range of the SC before returning the requested information. In our AS implementation, we set this maximum acceptable range to 20 meters.

4.2 Example Application: SocialAwareFlicks

We have implemented a context-aware mobile social networking application, called SocialAwareFlicks, which uses SSA and demonstrates the feasibility of the SSA framework. SocialAwareFlicks displays recommended movie trailers that match the movie preferences of one or more users jointly watching a common display. SocialAwareFlicks could be deployed in locations such as video-rental establishments for marketing purposes, to make customers aware of new video rentals that match their interests. The version of SocialAwareFlicks implemented using SSA is an enhancement of the previous version of the SocialAwareFlicks application described in [13], which used the SocialAware framework and did not consider security and privacy issues.

The SC in SocialAwareFlicks detects the presence of users watching a single common screen managed by the system and shows recommended movie trailers based on the favorite movie preferences expressed on each user's Facebook



Name	Author	Duration
Playlist		--:--
vlc_playlist54618.m3u		--:--
A Scanner Darkly		02:05
The Village		02:27
Revolutions Trailer	www.thematrix.com	02:30
Minority Report		01:20
Media Library		--:--

Figure 6. Screenshot of prototypes of the SocialAwareFlicks mobile and stationary components

profile. After detecting the presence of each nearby MC and obtaining the EID shared by each MC, the SocialAwareFlicks SC uses these EIDs to retrieve the Facebook movie preferences for each mobile user by accessing the appropriate web service on the AS. We assume that these movie preferences are expressed as a comma-separated list of the user's favorite movies. The SC then uses the Netflix REST API web service [14] to obtain a list of similar movies for each of the user's favorite movies. We assume that the user will prefer to watch movies that are similar to his/her favorite movies. After obtaining these recommendations for each user, the SC uses heuristics to create an aggregate playlist of movie titles that accommodates the preferences of the detected users in the group. [13] describes these playlist generations heuristics in some detail. Finally, the SC attempts to find a movie trailer file for each title on the playlist on the local filesystem, and begins playing the trailers on the screen using the VLC media player [27].

Figure 6 shows a screenshot of the current prototypes of the SocialAwareFlicks MC and SC. The MC for one user is running in the MicroEmulator [28] window, while the SC has generated a movie trailer playlist for this user and invoked VLC to begin playing trailers from this playlist. VLC is playing the trailer for the film "A Scanner Darkly".

4.3 Evaluation

Performance metrics were gathered for the SSA Authentication Server. All performance testing was performed using a Macbook Pro notebook running Mac OS X 10.5, with a 2.4 GHz Core2Duo processor, 4 GB of RAM, and a university-provided high-speed Internet connection.

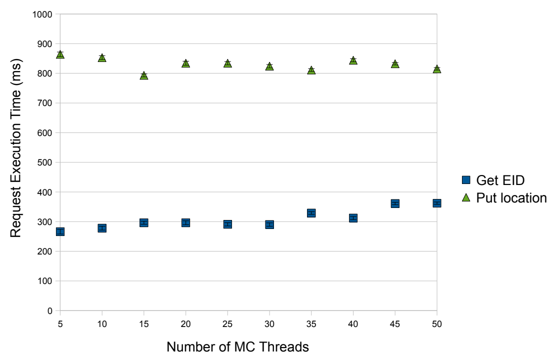


Figure 7. Scaling of SSA AS as the number of concurrent MC requests increases

Our AS performance test shows how the AS scales as the number of MCs in the system increases. For this test, an increasing number of MCs are configured to concurrently send requests to the AS. Each MC sends a HTTP GET request to retrieve it's current EID, followed by a HTTP PUT request to inform the AS of it's current location. We set each MC to maintain a constant throughput of 20 HTTP GET requests and 20 HTTP PUT requests sent per minute. These MC requests are simulated using the Apache JMeter [29] loading testing tool, and all requests are sent using HTTPS. Figure 7 shows a graph of the average execution time for each request as the number of concurrent MC request threads increases. In this test, each MC device is simulated by an MC request thread. We see from this graph that average execution time for each put-location request remains approximately constant as the number of concurrent MC requests increases, while the average execution time for each get-EID request increases gradually. These results show that the AS scales reasonably well.

We also measured the end-to-end processing time for SocialAwareFlicks SC favorite movie list retrieval process for a user, which is a measure of the time required for the MC to retrieve an EID from the AS, the SC to get an EID from the MC over Bluetooth, and the SC to then retrieve the list of the MC user's favorite movies from the AS. Our measurements of this end-to-end operation revealed a mean processing time of 1263 ms, with a standard error of 44.31 ms. This shows that the SC can retrieve the list of a mobile user's favorite movies reasonably quickly upon detection of an MC.

Note that all testing was performed in our local computing environment, so all data persistence communication between the AS deployment and SimpleDB traversed the Internet to reach Amazon's remote server data centers for SimpleDB. Thus, the round-trip latency between our AS deployment and Amazon's data centers dominated the processing time for each request sent to the AS. We expect that the time to process each request on the AS would be substantially reduced if we deploy the AS within Amazon's data centers using the Amazon Elastic Compute Cloud service [30].

5 Background and Related Work

This section will discuss the major motivating factors or changes in technology and society which drive this work. In particular recent developments in both mobile computing and social networks are discussed along with how these two areas are increasingly intersecting. Following this, a set of example applications which mix social networks with mobile computing are discussed along with a their associated security and privacy threats.

5.1 Mobile Computing

Advances in cell-phone connectivity and the ever-more ubiquitous smartphone have brought the Internet into the pockets of individuals anywhere, anytime. Along with the ability to always be "connected", social networks have developed resulting in very high penetration rates within certain communities (in particular college campuses). This means that within many social contexts there is a high number of social network users with mobile computing devices.

5.2 Social Networks

Online social networks have been exploding in popularity over the last few years. According to Facebook.com's statistics page, the site has over 200 million active users [3], of which over 100 million log on everyday. To compare this with ComScore's global Internet usage statistics [31], 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 country's Internet population (China is the largest with 179.7 million Internet users [31]). Facebook has recently become very popular among mobile 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.

5.3 Existing Mobile Social Network Applications

As mentioned previously, SSA builds on the authors' prior work on WhozThat [1] and SocialAware [13]. Both of these systems enable the creation of context-aware applications that exploit user information from social networking sites.

Most current approaches toward integrating social networks with mobile devices have missed the opportunity to bind social context tightly with the local context of interacting people. These approaches simply extend the Web interface of the social network to the mobile device, by providing a view of the user's social network on his/her mobile phone[32]. Recent work such as CenceMe has sought to enrich the amount of information flowing in the direction from the mobile device to the social network, e.g. the location of the user and perhaps context cues such as whether the person is talking [33]. While these cues provide more information about the user's context, they do not integrate social network information about nearby users with the user's local context. In contrast, the WhozThat system concepts used by SSA applications exploit 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, it populates its own database of social context information rather then connecting with popular online social networking sites, and does not consider the development of context-aware applications as discussed in this paper.

Commercial location-aware mobile social networking services, such as BrightKite [8] and Loopt [7], provide some of the functionality found in WhozThat and SSA. 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 SSA.

5.4 Privacy & Security

That mobile social networks present significant privacy problems is very apparent. Many projects have identified and presented solutions to some of these problems. Duke University's SmokeScreen project [5] uses "cliques" between users which are then resolved through a trusted broker system. Assuming that users know/trust each other they are then able to "sense" each other's presence. This project targeted the problems of snooping and power efficiency on mobile devices. Another related work out of Duke University is called "We Saw each other on the Subway" [6]. This paper chooses not to trust the central server system or other peers. It allows for users that were coincident in time and space to verify such a coincidence at a later time without compromising their "peer-to-server" or "peer-to-peer" anonymity. Peopletones [34] allowed for users to share their information through shared cell tower coverage. This provided a partial or fuzzy location sharing service. The SlyFi project [35] protected privacy through obfuscating even the identifier bits such as MAC address and other protocol fields that 802.11 would normally rely on to route packets. It then achieved reasonable performance without these identifiers, protecting the physical address of the user device.

6 Future Work

This paper provides a general architecture which connects users' social network information with location-based services anonymously. However, this architecture does not imply a specific implementation of any particular component. For instance EID's could be exchanged over different wireless/cellular protocols instead of Bluetooth. They could also be managed through a centralized infrastructure on the Internet or integrated into a peer-to-peer trust network. We intend to explore these alternative implementations in the future.

The way in which preferences are chosen and returned to the SC could be optimized for different metrics. In particular this mechanism could be designed to provide k-anonymity for a set of users EIDs rather than just one at a time. This problem relates to a more general set theory problem in which a set of social network information associated with a set of users is chosen such that the set of preferences cannot map back to any one or any set of the users within some guarantee. This problem may apply to more uses of social network information than just applications of the type described in this paper.

An area that we have chosen not to deal with in this paper is that of user-specified privacy controls on his/her social network data. It is understood that there are certain special cases for which a user may want certain information shared or not shared and more comprehensive tools must be developed to support this. Facebook has been developing tools for users to control their information at a finer granularity and these controls are respected by our system. However, we recognize that users may still feel that they do not have full control over how their information is used and this issue will continue to be important to the development of social networks in general, and mobile social networks in particular.

7 Conclusion

This work describes a novel framework to support an anonymous exchange of social network information with real-world location-based systems, allowing truly context-aware systems without compromising users' security and privacy. Secure SocialAware provides an information exchange solution which could support many different application models that currently require users to give full access to their identity and information. We hope that this 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.

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

[4] "Apple: 1 billion iphone apps," http://www.apple.com/itunes/billion-app-countdown/.

[5] 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.

[6] 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.

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

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

[9] "Facebook," http://www.facebook.com.

[10] "Myspace," http://myspace.com.

[11] “Whoshere overview,” http://myrete.com/whoshere.html.

[12] “Api - facebook developers wiki,” http://wiki.developers.facebook.com/index.php/API.

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

[14] “Netflix api,” http://developer.netflix.com.

[15] 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.

[16] 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.

[17] “Bluecove jsr-82 emulator module,” http://www.bluecove.org/bluecove-emu/.

[18] “Marge,” https://marge.dev.java.net.

[19] “The java community process program - jsrs: Java specification requests - detail jsr 82,” http://www.jcp.org/en/jsr/detail?id=82.

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

[21] “Restlet - lightweight rest framework,” http://www.restlet.org.

[22] “Rfc 4627: The application/json media type for javascript object notation (json),” http://tools.ietf.org/html/rfc4627.

[23] “Rfc 2617: Http authentication: Basic and digest access authentication,” http://tools.ietf.org/html/rfc2617.

[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] “Vlc media player,” http://www.videolan.org/vlc.

[28] “Microemulator,” http://www.microemu.org//.

[29] “Apache jmeter,” http://jakarta.apache.org/jmeter/.

[30] “Amazon elastic compute cloud (ec2),” http://aws.amazon.com/ec2/.

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

[32] “Blackberry facebook application,” http://www.facebook.com/apps/application.php?id=2254487659.

[33] 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.

[34] K. A. Li, T. Y. Sohn, S. Huang, and W. G. Griswold, “Peopletones: a system for the detection and notification of buddy proximity on mobile phones,” in *MobiSys ’08: Proceeding of the 6th international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2008, pp. 160–173.

[35] B. Greenstein, D. McCoy, J. Pang, T. Kohno, S. Seshan, and D. Wetherall, “Improving wireless privacy with an identifier-free link layer protocol,” in *MobiSys ’08: Proceeding of the 6th international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2008, pp. 40–53.