Towards a New Privacy-Aware Location Sharing Platform

Marcello Paolo Scipioni and Marc Langheinrich
University of Lugano (USI)
Faculty of Informatics
6904 Lugano, Switzerland
{marcello.paolo.scipioni, marc.langheinrich}@usi.ch

Abstract

Location-based social networking services are becoming increasingly popular among the multitude of mobile applications. The new location sharing functionalities made possible by GPS-based mobile phones, however, also raise two main privacy issues: users are typically limited to a only few built-in options that do not support fine-granular or even changing privacy preferences, and location data are often implicitly shared with service providers in the process. This paper argues for a new group-based architecture for privacy-aware location sharing, developed starting from a use-case analysis of sharing patterns and a brief review of existing privacy options in today's popular location-sharing applications. This is complemented by a stakeholder discussion with a view towards identifying the various location privacy threats. Based on this analysis, we present a decentralized architecture for location sharing, based on the popular XMPP instant messaging platform, which supports sharing location information at different granularities directly with different sets of contcts. This gives more flexibility to users in terms of sharing patterns, thus providing more privacy as users can decide in detail what kind of location data to share with whom. The system also avoid the central disclosure of all user location data to the service provider, by either employing private servers or relying on end-to-end encryption between contacts.

Keywords: Location Sharing, Location Privacy, Location Based Services, Mobile Location Privacy

1 Introduction

The widespread availability of GPS-enabled smart-phones has pushed the popularity of location-based mobile applications, which offer to share one's location with friends, to become the killer-apps of the moment. While most of these services are free, however, there is usually a price to pay: more often than not, a user's location is freely shared with marketers, who are interested in developing detailed location profiles to better target online advertising. Moreover, customers are usually only offered very basic privacy settings, limiting the sharing options of users to on-off decisions. Recent surveys [16] highlight that although people feel valuable this kind of services, users are concerned by the lack of control over location data in existing LBS. This approach, which forces users to make binary decisions, fails in giving effective ways to differentiate the level of detail (e.g. exact location or neighbourhood, city, or country level location) of the shared data among different friends, making the service less versatile in some situations.

Ideally, location sharing applications would support the rich sharing preferences of their users, while limiting location disclosure to all but the intended recipient. In this article we present our current location sharing prototype system, *LoSha*, which attempts to address these aspects and ultimately serve as a research vehicle to explore actual user sharing behaviors.

As a first step in the development of *LoSha*, we enumerate the main use case scenarios for location sharing in Section 2, focusing on the user needs and on the sharing patterns taking place in each scenario.

Journal of Internet Services and Information Security (JISIS), volume: 1, number: 4, pp. 47-59

Section 3 then looks at four existing location sharing applications, namely Google Latitude, Facebook Places, Glympse, and Microsoft's "Bing – We're In". For each application, we highlight the sharing patterns and privacy options offered, and discuss how suitable its structure is to serve the presented scenarios. Section 4 discusses the role played by different stakeholders and evaluates possible privacy threats in light of the interaction among the stakeholders.

After the above analysis, Section 5 presents our privacy-aware architecture that aims at enriching the location sharing controls today available in real applications, while at the same time providing some baseline privacy protection from service operators. We describe two initial prototypes that focus on each of the two aspects in turn, and outline ideas for merging them in a unified architecture. Section 6 then closes with a summary and an outlook on a field study we plan to undertake in the next few months with the first completed prototype.

2 Location sharing use case scenarios

The ubiquitous availability of location based services, favoured by the widespread diffusion of modern smart-phones, has enabled a number of functionalities which were impossible before the mobile revolution. Location sharing applications have become a useful tool in many situations: meeting an old friend who happens to be in the neighbourhood, checking if one's partner is still at office and asking to buy something at the supermarket on the way home, or letting colleagues know that one is running late for a meeting. These examples in fact map to the three main use cases we see for location-sharing applications:

1. **Serendipitous encounters:** Probably the most-often advertised feauture of location sharing services is that it allows for spontaneous encounters: Each user informs the system of his or her current position, which allows the service to alert users that one or more of their friends are in their immediate vicinity. A user may of course also actively query for people in the vicinity, which simplifies the system as no "alerting" rules need to be specified. Instead of having it constantly updated in the background, a user may also explicitly announce his or her current location, which implicitly indicates that one is open to meet others. However, the basic effect of all three of these approaches is that people can "run into each other" much more frequently.

2. **Rendezvous:** A common problem when actually planning to meet others is that thing seldom go according to plan. A traffic jam may occur, a train may be delayed, or something comes up at the last minute. A location sharing application can allow one to see at a glance where others are and when they are expected to arrive at a particular meeting point. With more than two people meeting, this approach can be much more efficient that manually calling up each group member and asking them information about their whereabouts. A fully automated system may again alert users if some of (ad-hoc) meeting members are running late, though a simple query screen showing everybody's current position would achieve the same effect. In this scenario, explicitly announcing one's position would also work: the first to arrive at a pre-agreed spot could simply push a "I'm here first!" button, letting the other members know he or she is waiting, while someone running late could hit a "Sorry, running late!" button that would inform all other members about the person's current position and – if the meeting place is known to the software – the estimated time of arrival.

3. **Staying connected:** Apart from the more functional use cases above, people may also want to share their location with others in order to stay "in touch" with friends. This works well because location is often strongly linked to an activity, especially for people who know each other well,

(a) Google Latitude (b) Facebook Places (c) Glympse

Figure 1: Screenshots of existing location sharing applications.

such as close friends or family members. Being in a bank building may mean for some to be still at work, while for others it might mean that they are already on their way to dinner and are simply stopping to get some cash. Consolvo et al. [9] showed how the resulting social coupling is used by household members to implicitly interact with each other: a wife may notice that her husband is still at work and thus spontaneously text him to pick up something from the dry cleaner on his way home. Or a student studying abroad may share his daily routine with friends back home – not with a view towards meeting them, but in order to maintain their social relationship.

Ideally, location sharing applications support users in all of the three use cases, while minimizing the chances of unintended or unnecessary location disclosures. To understand the design space better, we first look at existing location sharing applications (Section 3 below) and examine their respective privacy controls.

3 Existing services and sharing models

The three use cases described above are covered to various extents by existing location sharing applications. Below we compare a few popular applications with respect to their coverage of these cases, and describe their support for location privacy. Specificially, we will focus on the following three services: Google Latitude, Facebook Places, and Glympse. Although many other applications are available on market today, we consider the sharing models and the privacy models adopted by these services as a representative sample of the privacy models offered these days by location sharing applications, due to their popularity, but also because many other services offer similar functionality and/or provide similar privacy controls.

3.1 Google Latitude

Latitude is a location-based mobile service offered by Google for sharing location with friends [5]. Latitude offers a user interface (UI) showing a map where users can see their position and their friends' ones overlayed onto the map, as depicted in Figure 1(a). Users can invite their friends who have a Google account to join their network of contacts and share their location. Latitude offers two levels of controls:

privacy settings and sharing options. Through the privacy settings, each user can decide how his location is reported to the Latitude service. Three options are available:

- "Detect your location", which allows Latitude to know the user's best available location on user's device (GPS, cell-based or IP-based location);
- "Set your location", which allows users to manually choose a location by tapping on the map or typing an address, ignoring any automatic positioning tools available;
- "Do not update your location", which blocks any location disclosure to the Latitude service.

The location updates, collected by Latitude according to the general privacy settings above, can be further filtered on a per-user basis choosing one of the available sharing options:

- "Share best available location";
- "Share only city level location";
- "Hide from this friend".

Moreover, Latitude allows users to "check-in" into specific places previously determined by the user, either manually or automatically, when the user appears to be in one of the designated locations.

Latitude allows users to track themselves over time, but does not support sharing this aggregated/historic data. The information shared with others is always the last known position, i.e., it does not support sharin tracks or movements.

3.2 Facebook Places

Another widely used location sharing service is Facebook Places, an extension to the popular Facebook application which is available to Facebook users [2]. Places is based on the "check-in" paradigm: users can select their current location from a list of existing nearby places, or add a new location and its description (cf. Figure 1(b)) and enter in the virtual representation of the selected place. After checking in, users can share their location as any other piece of information in Facebook, through the existing network of friends. Other similar check-in based location sharing applications are Brightkite [1], Foursquare [3], Gowalla [6], Loopt [7], and Whrrl [8].

Privacy options are offered to Facebook users for managing the disclosure of places to friends: the places where a user checks-in can be set visible only to the same user, to friends, friends of friends, everybody, or to specific people which then need to be listed explicitly, following the general privacy model to which Facebook users are familiar with. Moreover, users can check-in together with their friends just by tagging these friends at check-in time. This means that other people can change the current location of another user (which clearly has to be a friend), if the latter has enabled this option.

3.3 Glympse

A location sharing app employing a different location sharing paradigm is Glympse [4]. With Glympse, people can share their location for a limited amount of time – typically between 15 and 30 minutes – sending a so-called "Glympse", i.e. a location in the form of a web link. There is no need for an account, neither users are supposed to specify any contacts; simply, the Glympse can be sent per email or as a text message. The Glympse sent by a user can be viewed by the recipient either inside the UI of the Glympse application, if he is a Glympse user as well – see Figure 1 (d) – or in any web browser. The location will be updated from the sender's device and will be displayed on a map, together with the current speed and

the remaining sharing time, until it expires. Obviously, the sender can decide to manually stop the current Glympse before the time specified in advance. There are no other specific privacy countermeasure, other than the (usually short) expiration time: the recipient could share the Glympse with others, since it is simply a web link; obviously the location updates will be visible only until the Glympse expires. A very similar approach is also taken by Microsoft's "Bing – We're In" app, which was released in August 2011.

3.4 Sharing Models and Use Cases

Although all the described applications let users share their location, each of them is different in the sharing patterns offered and in the way location is shown through the UI, which make the resulting application more suitable for some uses and less for others.

In Facebook Places, for instance, a user's location is represented by an item (it can be a shop, a disco, a bar, a concert, etc...), through a symbolic representation, rather than by a point moving on a map, i.e. a set of spatio-temporal coordinates. The symbolic representation of location data favours the use of Facebook Places mainly as a tool for rendezvous and meetings, while it is less suitable for pushing quick "I'm late" messages. On the other hand, Glympse is perfect for the "I'm late" scenario, since one can rapidly choose from a list of pre-defined messages and send it together with his location updates to any of the phone contacts as a text message, or by email, without needing the recipient of being subscribed and logged into the service. Although no log-in means versatility while sending quick messages, it is however impossible to get location from friends outside the (short) duration of a Glympse, making this service less suitable for the "serendipity" and "stay connected" scenarios.

Google Latitude, on the other hand, can be a helpful tool in the above scenarios, because the location of all currently on-line friends are shown on the UI after launching the application. Moreover, the check-in feature makes Latitude a useful tool also in the rendezvous scenario. The main drawback of a service where location is shared for indefinitely long periods of time is the management of visibility and location granularity for all contacts: it seems reasonable to assume that "stay connected" scenarios can be applied to a very limited subset of contacts, as well as the "serendipity" scenario makes sense only for contacts which are in the same city (or quarter), while for other contacts a lower location granularity level (city or country level) could be employed. The lower location granularity, apart from granting more privacy to the user, would not interfere with the functionalities of the mentioned scenarios. There is however no easy way to manage these features without keeping changing manually the sharing options.

With Glympse, users can explicitly limit the time for which location sharing is active. This limiting addresses privacy concerns well, but on the other side it also limits the use of this service in the "serendipity" scenario. Microsoft's *We're In* additionally allows users to see which of their contacts are already sharing location before deciding to start sharing theirs; this makes *We're In* slightly more suitable for the rendezvous scenario. In contrast to Google Latitude, which does not support the sharing of historic data, all sharing instances in Glympse share the entire track of a person, i.e., the recipients not only see where the tracked person is, but also where the person went from the time the sharing session started. This functionality does not seem to help in the rendezvous scenario, though could be beneficial in a staying-connected scenario, for which Glympse is ill-suited, however.

4 Location Privacy and Stakeholders in Location Based Services

Privacy is an issue to be taken into account not only because of security reasons. If people start to share, it does not necessarily mean that people want to share everything with everybody. As Dourish and Anderson [10] explain: "Privacy is not simply a way that information is managed but how social relationships are managed". To this end, a study by Consolvo et al. [9] was conducted explicitly asking

people to disclose their location data. It turned out that the most important factors that people take into consideration before disclosing their location data are: *who* is requesting location data, *why* he needs it and at *what level* of detail. The implication of privacy with social relationships has therefore to be carefully considered, and for this reason location privacy strategies should take it into account.

The easiest way to apparently "solve" location privacy is to manually or automatically authorize (or not) the disclosure of location information to others. As a result, sharing location data would lead to a binary decision: friends would receive full access to location data, and strangers would be blocked. This is in fact the most basic protection that services like Google Latitude offer: one can easily share and "unshare" one's location information with individual users. This approach, however, quickly becomes limiting, as it forces people to choose between "on" and "off", between "black" and "white", without considering all those "grey" levels that a dynamic privacy negotiation process usually involves [10]. Consequently, more complex and/or powerful methods have been proposed.

Privacy can be described as "the desire of people to choose freely under what circumstances and to what extent they will expose themselves, their attitudes and their behaviour to others" [21]. Safeguarding location information is just one of the many "data points" that make up the attitude and behavior of people, yet it is a particularly powerful one, as a place is often tightly connected to an activity (e.g., a shopping mall, an office), an interest/belief (e.g., a church, a political rally), or a personal attribute (e.g., a prison, a clinic).

As with any privacy assessment, one of the most basic question is: what "privacy-risks" should the system safeguard against [14]? In location-based service scenarios, one can differentiate between three primary actors:

- *Intended recipient*, e.g., the taxi company to send you that cab, or your mom so that she knows where you are. This usually involves the use of a *service provider* that offers to forward your location to the intended recipient.

- *Service provider*, e.g., Google providing you with the *Latitude* application, or a restaurant recommendation system for near-by places. In contrast to the *intended recipient*, users usually do not have a primary goal of letting the service provider know their location – it is a by-product of getting a restaurant review or staying in touch with friends.

- *Infrastructure provider*, e.g., your mobile phone company, or the operator of an indoor location system. While self-positioning systems such as GPS can work without an infrastructure provider, mobile phone users are often implicitly located in order to provide communication services (i.e., route phone calls).

In addition, a location-based system might have a number of unintended recipients, such as:

- *Accidental recipient*, e.g., your parents when you claim to be at a friend's place studying, but forgot to turn off the system when you went out drinking.

- *Illegal recipient*, e.g., a hacker intercepting your wireless location updates, or breaking into the service provider's or the infrastructure provider's records.

- *Law enforcement*, e.g., police or other government agencies accessing the service provider's records.

Clearly, for a given system, the *infrastructure provider* needs to be trusted, as location information is not available otherwise. While GPS alleviates the need for a provider, mobile phones are always traceable through their connection to a particular antenna. Obviously, withholding location information from the *intended recipient* seems not very useful. Both *illegal recipients* and *law enforcement* are best controlled by leaving as few information as possible stored on external servers – laws that limit data

retention periods might help here, as would a service architecture that would minimize data collection on those servers in the first place. *Accidental recipients* will require appropriate control tools (manual and/or automated) that can support users in minimizing such accidental disclosures. Today's *(location-based-) service providers* often offer the service for free in exchange for tracking the user's location – a technological solution withholding location data from them would need to be complemented with an appropriate alternative revenue model.

5 Message-based Privacy-Aware Location Sharing

Location privacy in a location sharing service cannot be provided just as an optional add-on feature to plug into an existing service. A location sharing service can be privacy-aware if it is designed to offer privacy through the sharing patterns offered, in a "privacy by design" approach. Below we present our prototypical architecture for providing a location-sharing service that supports the three main use cases described above, while minimizing privacy risks of unwanted location disclosures. In particular, we target three main requirements:

- **Provider-less sharing:** All currently available applications typically require the service provider to know the location of all users at the maximum available level of detail. In Latitude, for instance, after uploading location to the service provider at the most accurate granularity, it is possible to lower the level of detail of the location sent to friends. In principle, however, it is not required that the service provider has any knowledge of location data for the purpose of sharing location with other users. The most relevant privacy threat which users are exposed to is in fact the collection of location data from the service provider for detailed users profiling. Our architecture thus attempts to decouple the service provider from the actually shared location data. Note that this might require different business models from today's prevalent free apps, where the service provider's revenue comes not from user fees but from marketing of (typically aggregated) location data of its users.

- **Group-based sharing:** As remarked in Section 4, privacy must take into account the dynamics of social relationships; a privacy-aware location sharing service, therefore, has to offer the chance to differentiate the degree of detail of the shared location data, in the same way as in real life people share things with different people at a different level of detail. As we pointed out in Section 3, services which let users share location without time constraints are more suitable for the "serendipity" and the "stay connected" scenarios. We propose to augment this use-case with a group-based approach. Once contacts are divided into groups, users can share location at different granularities with different groups, and/or share at different times location with different granularities. Supporting different sharing patterns on a group basis not only lets us offer a more versatile tool, but also goes in the direction of usability: an application which only lets users change sharing patterns at a user level would hardly be maintained with hundreds of contacts (which today seems to be an average number). The subdivision of contacts into groups has also the goal of facilitating users in the differentiation between intended recipients and accidental recipients, to which different sharing patterns can be applied. This approach is similar to Google's recently launched social networking service Google Plus, which allows one to arrange contacts into "circles".

- **Agile sharing:** To support the "rendezvous" use case of one-off sharing scenarios, the UI must make it very simple to quickly share one's location with others, without the need for grouping the recipient(s) first or even requiring the recipient(s) to have a corresponding application installed. The above-discussed Glympse is a good example of such a service, as it does not require recipients to have the app installed, nor does it require (nor support!) grouping of contacts. Our service must make it easy for users to quickly share their location for each of the three use cases discussed.

Below we present two individual architectures that demonstrate various aspects of the final system that we are currently working on. We will also present the current prototype application and discuss our deployment plans.

5.1 Provider-less Sharing through Encryption

A simple way to avoid disclosing location data to the service provider is through encryption. Our first experiment consists of a simple messenger architecture with a client-server structure. We based the architecture of our application on XMPP, an open-source protocol widely used for instant messaging and user presence notification[1] [22]. After authenticating to the server and logging into the service, users can add contacts to their roster (i.e. a list of contacts, according to the XMPP terminology) and start sharing their location with on-line friends. The clients start exchanging messages containing their GPS position at periodic time intervals which can be adjusted by users. The messages are then received by the recipient clients, and the avatars of the senders are displayed at the specified positions on a map. The whole traffic is encrypted end-to-end with public key cryptography: the clients can send encrypted location updates, and the server only works as a router to send encrypted messages to the recipients, as shown in Figure 2.

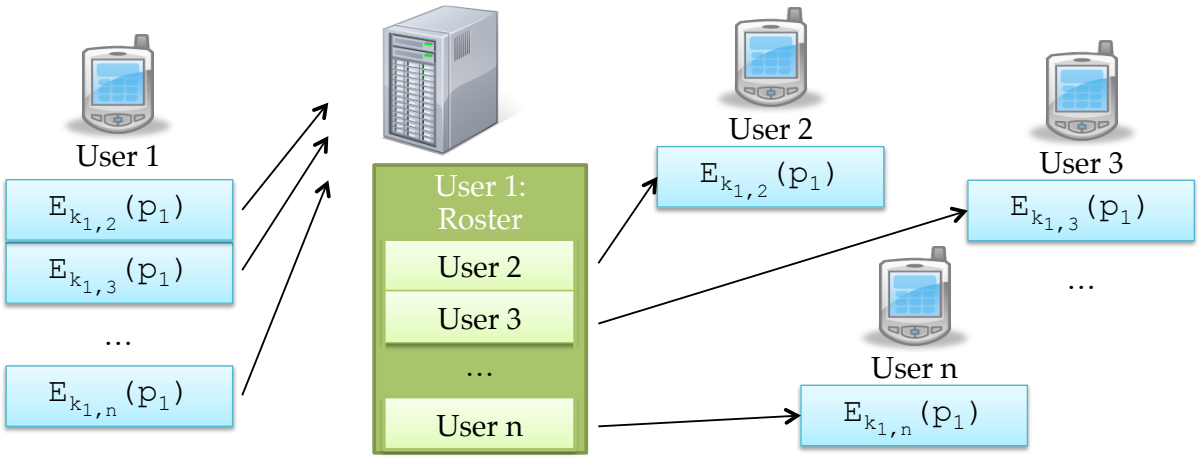


Figure 2: Architecture of the prototype direct location sharing application: User 1 shares his location p_1 with his friends after encrypting it with a different asymmetric key for each friend $k_{1,2}, \ldots, k_{1,n}$.

This prototype system consists of a client application developed for the Android environment and an open-source implementation of XMPP server from the server side, called Openfire [13]. The location information is read locally from the GPS module of the phone encoded by clients according to the Geoloc format[2] and embedded into the messages sent to friends. The presence notification feature, offered by the XMPP protocol, lets the clients know which friends in the roster are currently logged in. Each client then notifies the on-line contacts with encrypted location updates. This approach gives full control over location data which are handled by the clients directly, without disclosing them to any external repository.

With this configuration all the location data exchanged among users are hidden to the service provider, which just offers the infrastructure for the exchange of encrypted data. Location notifications, however, can only take place when both users are simultaneously online. Moreover, each client needs to prepare as many messages as the number of friends to notify, since each message has to be encrypted with a different key for each friend (see Figure 2).

[1] XMPP is used e.g. in Google Talk (see `www.google.com/talk`) and Facebook chats (see `www.facebook.com/sitetour/chat.php`).

[2] The Geoloc format is specified at `http://xmpp.org/extensions/xep-0080.html`

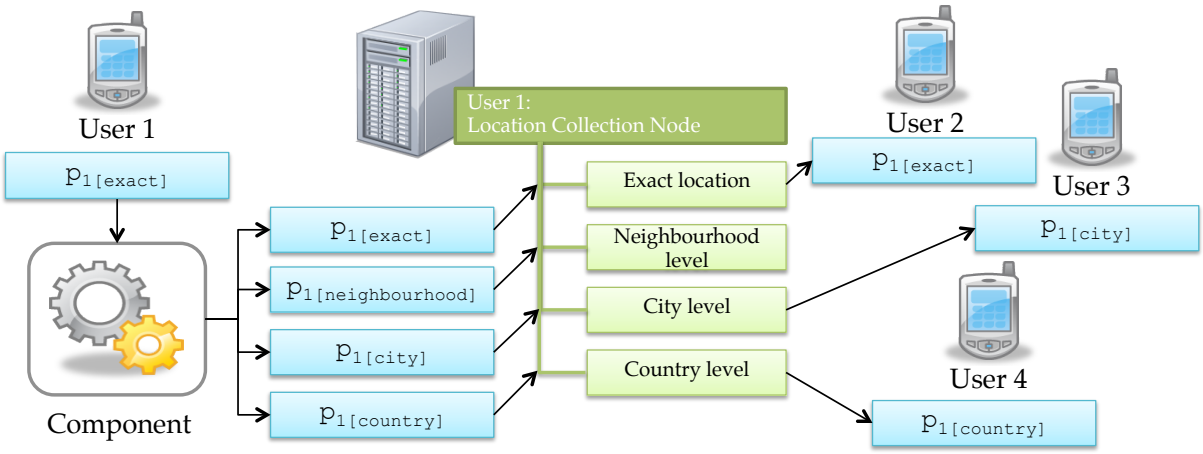


Figure 3: Architecture of the group-based location sharing platform: the client sends a single message to a server-side component, which updates all the pubsub nodes according to their accuracy.

Furthermore, due to the decentralized nature of the XMPP protocol, the single central server can be substituted by a network of XMPP servers; bringing it to the extreme, it is possible to use one XMPP server per user, connected to others in a peer-to-peer fashion. The absence of a single server can lower the privacy risks associated to the centralized structure, while the performance analysis conducted by Mayrhofer et al. [15] showed that the fully peer-to-peer approach is still feasible.

5.2 Group-based Location Sharing with PubSub

The simple architecture presented above protects users from unwanted location disclosure to the service provider; however, it does not provide any way for differentiating the location information to share with different contacts, neither it offers functionalities to arrange multiple contacts into groups. We replaced the messenger structure described above with a publish-subscribe (pubsub) client-server architecture, which allows to share location at various granularities (e.g. exact location, neighbourhood, city, or country level location), and where the privacy options are enforced on a group basis. For this architecture we assumed a very simple centralized pubsub model, in which clients directly contact the server without involving intermediate brokers.

The general structure of the conceived architecture, which is still based on the XMPP protocol, is shown in Figure 3. After authenticating to the XMPP server, each user can add new contacts to his roster through the client application. All the contacts in the client application are organized into groups, to represent the social relation established with the group members (e.g.: family, close friends, workmates, volleyball team, etc...). Instead of sharing location with individual users, here location is shared with entire groups. For each group the location granularity can be set, so that all group members can receive location notifications with the granularity set for the group.

From the server side, each user is associated with a *collection node*, i.e. a set of pubsub nodes, where each pubsub node corresponds to a group of contacts created by the user in the client application. A *component* works as a trusted third party among the client and the server[3]. For each new group of contacts, the component creates a new pubsub node, containing location data with the accuracy specified by the originating user. The client application never pushes data directly to the pubsub nodes; rather,

[3]The communication protocol between the XMPP server and the component is specified at `http://xmpp.org/extensions/xep-0114.html`

it sends them in a packet to the component which parses it and reverse geocodes it[4] in order to retrieve the strings to describe the current location – which is defined by latitude and longitude values – also for coarser granularities. Finally, the component pushes data onto different pubsub nodes according to their granularity. Contacts can read data only from nodes corresponding to groups they are member of, exploiting the white-list mechanism for pubsub offered by the XMPP protocol. When a new granularity is set for a group, the location published in the corresponding node is updated accordingly, and all the contacts subscribed to that node will receive the location with the new granularity.

An advantage of this approach is that the client application has to send just one message to the component, and all the friends can see the location update through the pubsub node they have access to. Moreover the pubsub mechanism also enables cashing of location data when users are off-line, making always available on the server the latest location update. Note that in this model, however, the XMPP server receives location information in the clear, as the data cannot be encrypted with just a single user's key anymore.

5.3 A First Implementation: LoSha

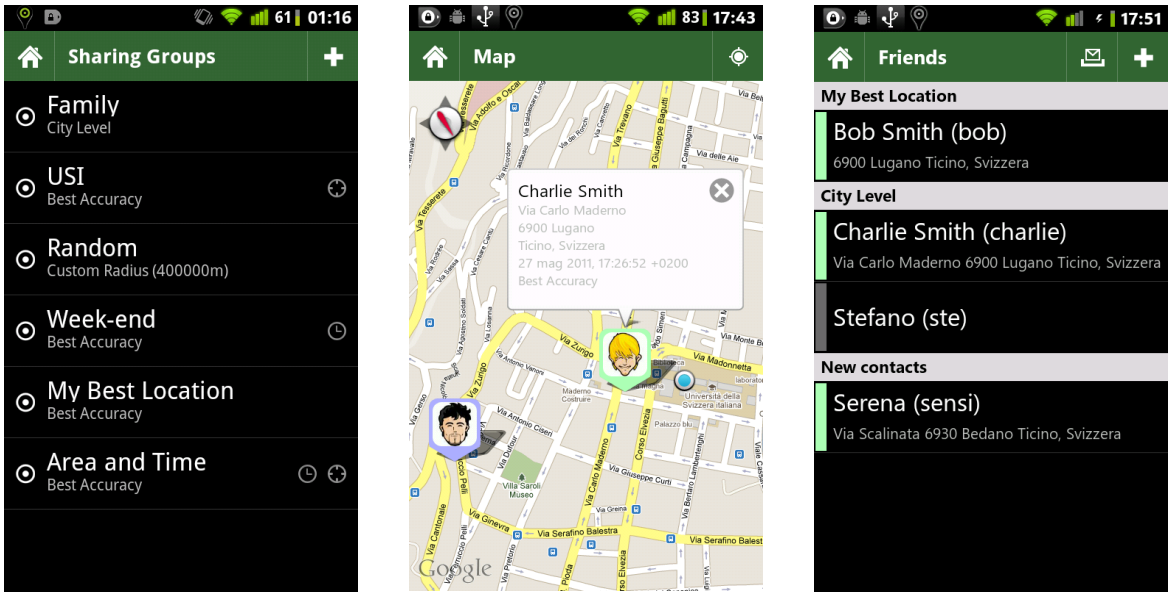
We have implemented a prototype application with the group-based location sharing architecture described above. Groups of contacts can be created, to arrange them according to the existing social relations of a user. When a group is created, a location granularity is set for the new group: all the members of the group will receive location updates with the specified granularity. All the groups created by a user are shown in the "Sharing Groups" screen of the prototype, shown in Figure 4 (a). Below the name of the group, the location granularity chosen for that group is displayed. All the friends with whom the location is shared are shown in the "Friends" screen, as shown in Figure 4 (c). The contacts appear in this screen sorted by sharing group; furthermore, the location of every contact is displayed as an address in this screen, as well as his presence (or absence) online, represented by a green (or grey) marker. The map screen of the application displays the positions of friends with the common representation of avatars on a map, as shown in Figure 4 (b).

Location can be shared with the members of a group not only according to different granularities, but also according to rule-based location disclosure: the group sharing settings can be configured in such a way to limit the location disclosures within user-defined time slots and/or when the subject is within a certain area, similarly to the privacy settings available in Loccaccino [20].

The current implementation of the system is focused on the group-based sharing patterns, with the goal of building an initial prototype solid enough for running a field study in which real user behaviour is analysed and the new functionalities can be evaluated. This work is however still in the exploratory phase, and several different directions can be taken.

In the next phases of development we plan to enrich the prototype by re-integrating encryption into the pubsub architecture. To cope with the dynamic nature of groups, group-based cryptography [18] could be used, as specialized group key management would be needed to handle revocation and group modifications. This additional feature would allow the system to protect from unwanted location disclosures towards the service provider, while at the same time offering more versatility in the sharing patterns. Moreover, due to the modular architecture of the system, a number of future architectural extensions are possible. In particular, the data leakage towards the reverse geocoding service provider should be avoided: we are planning to implement a reverse geocoding service directly on the client device using publicly available geo data. Datasets with city, region and state boundaries can easily be loaded onto the large storage cards of modern smartphones, while the actual reverse geocoding process

[4]The reverse geocoding service available through the Google Maps API is used.



(a) Groups of contacts.

(b) Map view.

(c) Friends listed in the groups they belong to.

Figure 4: Screenshots of the prototype group-based location sharing application *LoSha*.

is something today's dual-core processors can afford without too much computational effort. [5] The perspective of this work is to allow for automatic disclosure of location with the appropriate accuracy: the system would need then to compute the distance among two encrypted locations and consequently decide the right accuracy for the current friend. Homomorphic encryption [11, 17, 12, 19] could be used in order to outsource this step to the service provider, so that distances can be computed blindly, i.e., without needing to decrypt any location data.

6 Summary and Outlook

We have presented in this paper a new architecture for location sharing which enforces the sharing patterns on a group basis. The group-based solution offers from one side more elasticity to users – which can share location at different granularities with contacts belonging to different groups – while at the same time it is designed to safeguard privacy and minimize unwanted location disclosure towards unintended recipients, helping users in the subdivision of contacts according to different sharing patterns through the group structure. Our proposed solution for a privacy-aware location sharing architecture is based on the analysis of user needs, elicited through use-case scenarios, as well as on a stakeholder analysis which takes into account all the subjects which take part in location based services and the privacy issues arising from the interactions among them. We plan to test the described prototype application with real users to verify the actual use of the conceived location sharing solutions by real people.

The outlook of this work is to enrich our first prototype by offering automatic location granularity management. This feature will allow users to automatically disclose more detailed location data with nearby friends, based on the distance among each other, while only coarse-grained location will be disclosed to far away friends. Apart from the distance-based location disclosure, more features should be

[5] Datasets with Swiss boundaries can be found, e.g. at `www.swisstopo.admin.ch/internet/swisstopo/en/home/products/landscape/swissBOUNDARIES3D.html`

added, such as the end-to-end encryption of all location data. The resulting architecture will be able to protect against unwanted disclosure of location data to the service provider and to give more versatility in the sharing patterns through the automatic differentiation of location granularity. Such automated rules for managing group-based sharing can be of help in scenarios where location is shared for long time and the level of detail of shared data depends on the distance among users and on the social relationships among each other. This may ultimately allow for fluid changes between the three described use cases, as the granularity of the data moves from "stay connected" (coarse, long-standing) to "serendipity" (city-level, chance encounter) and "rendezvous" (detailed, if close-by) scenarios.

Acknowledgments

This work was developed within the PALS project, funded by the Swiss National Science Foundation under grant n. 200021_129674. The authors would like to thank the anonymous reviewers for their valuable comments.

References

[1] Brightkite. `http://www.brightkite.com/`. Accessed September 2011.

[2] Facebook Places. `http://www.facebook.com/places`. Accessed September 2011.

[3] Foursquare. `https://foursquare.com/`. Accessed September 2011.

[4] Glympse. `http://www.glympse.com`. Accessed September 2011.

[5] Google Latitude. `http://www.google.com/latitude`. Accessed September 2011.

[6] Gowalla. `http://www.gowalla.com/`. Accessed September 2011.

[7] Loopt. `http://www.loopt.com/`. Accessed September 2011.

[8] Whrrl. `http://www.whrrl.com/`. Accessed September 2011.

[9] S. Consolvo, I. E. Smith, T. Matthews, A. LaMarca, J. Tabert, and P. Powledge. Location disclosure to social relations: why, when, & what people want to share. In *Proc. of the 2005 SIGCHI conference on Human factors in computing systems (CHI'05), Portland, Oregon, USA*, pages 81–90. ACM, April 2005.

[10] P. Dourish and K. Anderson. Collective information practice: Exploring privacy and security as social and cultural phenomena. *Human-Computer Interaction*, 21(3):319–342, 2006.

[11] C. Fontaine and F. Galand. A survey of homomorphic encryption for nonspecialists. *EURASIP Journal on Information Security*, 2007(13801):1–15, 2007.

[12] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Proc. of the 41st annual ACM Symposium on Theory of Computing (STOC'09), Bethesda, Maryland, USA*, pages 169–178. ACM, May-June 2009.

[13] Ignite Realtime. Openfire. `http://www.igniterealtime.org/projects/openfire/`. Accessed September 2011.

[14] M. Langheinrich. Privacy in ubiquitous computing. In J. Krumm, editor, *Ubiquitous Computing*, pages 95–160. CRC Press, Sept. 2009.

[15] R. Mayrhofer, C. Holzmann, and R. Koprivec. Friends Radar: Towards a private P2P location sharing platform. In *Proc. of the 1st International Workshop on Mobile Computing Platforms and Technologies (MCPT'11), Las Palmas, Gran Canaria, Spain*, February 2011.

[16] Microsoft. Location & Privacy: Where Are We Headed? Data Privacy Day 2011. `http://www.microsoft.com/privacy/dpd/default.aspx`. Accessed September 2011.

[17] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proc. of the 17th International Conference on Theory and Application of Cryptographic Techniques (EUROCRYPT'99), Prague, Czech Republic, LNCS*, volume 1592, pages 223–238. Springer-Verlag, May 1999.

[18] S. Rafaeli and D. Hutchison. A survey of key management for secure group communication. *ACM Computing Surveys (CSUR)*, 35(3):309–329, 2003.

[19] S. D. Rane, W. Sun, and A. Vetro. Secure distortion computation among untrusting parties using homomorphic encryption. In *Proc. of the16th IEEE International Conference on Image Processing (ICIP'09), Cairo, Egypt*, pages 1485–1488. IEEE, November 2009.

[20] E. Toch, J. Cranshaw, P. H. Drielsma, J. Y. Tsai, P. G. Kelley, J. Springfield, L. Cranor, J. Hong, and N. Sadeh. Empirical models of privacy in location sharing. In *Proc. of the 12th ACM international conference on Ubiquitous Computing (Ubicomp'10), Copenhagen, Denmark*, pages 129–138. ACM, September 2010.

[21] A. F. Westin. *Privacy and Freedom*. Atheneum, 1967.

[22] XMPP Standards Foundation. XMPP Protocol. `http://xmpp.org`. Accessed September 2011.

Marcello Paolo Scipioni is a Ph.D. student and a research and teaching assistant at the Faculty of Informatics at the University of Lugano (USI), Switzerland. Marcello obtained his M.Sc. in Computer Engineering from the Politecnico di Milano, Como Campus, in 2009, in the track of Sound Engineering and Design. His current research interests cover location privacy, information security and human-computer interaction. Marcello works on the Privacy-Aware Location Sharing (PALS) project, funded by the Swiss National Science Foundation (SNF).

Marc Langheinrich is assistant professor for Computer Science at the University of Lugano (USI) in Switzerland, where he heads the Research Group for Ubiquitous Computing since September 2008. Marc received his PhD (Dr. sc.) on the topic "Privacy in Ubiquitous Computing" from the ETH Zurich, Switzerland, in 2005. Marc is one of the authors of P3P, a W3C-standard for privacy on the Web, and has published extensively on privacy aspects of ubiquitous and pervasive computing systems (3000+ citations in Google Scholar). Marc is a regular program committee member of various conferences and workshops in the areas of pervasive computing, security and privacy, and usability. Marc is a Steering Committee member of many major conferences in the area of ubiquitous and pervasive computing, such as INSS, Pervasive, UbiComp, and PerCom. Marc also currently serves as the Steering Committee Chair for the Internet of Things (IoT) conference series.