Hangout: A Privacy Preserving Location Based Social Networking Service.

Murali Annavaram,
Ming Hsieh Department of Electrical Engineering,
University of Southern California, Los Angeles.
annavara@usc.edu

Quinn Jacobson
Nokia Research Center,
Palo Alto.
quinn.jacobson@nokia.com

Abstract

As mobile devices enter a new era with high speed connectivity and increasing compute capabilities a new class of applications called social networking applications is being showcased as the next revolution in mobile computing. In this class of applications each user in a social group contributes their knowledge about their surrounding environments and the collective knowledge can then be exploited by the group members for a personal or social benefit. As the popularity of mobile social networks increases there is a growing realization that information collected about an individual user can compromise one's privacy and potentially security. It is in this context we developed HangOut a privacy preserving social networking application. Hangout protects user's private information not only from other malicious users but even from system administrators who may have unrestricted access to the backend server that is providing the social networking service. Hangout uses location and time distortions, symmetric key encryption where the keys are exchanged in a peer-to-peer fashion and several client side controls to aggressively protect privacy with minimal degradation in user's perceived service value.

1. Introduction to Device Variations

The compute capability of today's high end mobile devices rivals the desktop performance of the early 90s. Thanks to Moore's law growth in transistor count these devices also integrate a rich set of environmental sensors such as GPS receivers, high resolution video cameras. As these feature rich devices ubiquitously connect us to the digital world they continuously collect and store information that pertains to the user. The information could be user location collected through GPS readings, pictures taken by the user from the cell phone camera. Some of the information stored will become sensitive information that relates to user's movement history in the form of GPS track. Applications and service providers rely on accessing this sensitive information to provide useful services. One prominent application is location based services that use the GPS capability to provide information that is relevant to a user's location. While location information is necessary for providing these services, these devices can potentially become tracking devices if the location information is continuously revealed to the application service providers. As the popularity of these applications grows information privacy is a cause for serious concern.

In response to growing privacy concerns several mobile service providers have published their privacy policies. These policies are intended to restrict how the information collected from mobile users will be used. For instance, the policy guidelines include enforcing data access control where only a limited number of people or software modules are given access to the user location information. However, such policy guidelines alone have been shown to be ineffective either because the data integrity is compromised by hackers or because of inadvertent disclosure of information to those who don't need access to such a data. The social expediency of preserving privacy in mobile environments can not be emphasized enough if mobile applications have to continue to flourish.

This paper presents HangOut a new location based mobile service prototype developed at the Nokia Research Center in Palo Alto that focuses exclusively on preserving location privacy of the user while still providing location relevant information. More specifically, HangOut is a social networking application that allows its users to interact with each other without *ever* disclosing the precise location information of any single user. HangOut preserves participants' privacy in two fundamental ways. It allows users to control their privacy while sharing their location with other participants. It also protects users against having their location tracked by anyone, including the service provider itself. HangOut ensures that there is no record of where users where that can be recreated from the system.

The rest of this paper is organized as follows. Section 2 describes the HangOut prototype and the various usage models that are implemented in the prototype. Section **Error! Reference source not found.** presents

how privacy is preserved in HangOut in these usage models. Section **Error! Reference source not found.** presents a brief survey of the related work and we conclude in Section 5.

2. HangOut Usage Models

HangOut is a mobile service that lets participants share their location information while preserving their privacy. The HangOut prototype implementation envisioned three usage scenarios in the context of social networking. This section describes just the usage scenarios without detailing how privacy is preserved in HangOut. Section **4** will then present the details on how privacy is preserved in each of these usage scenarios.

2.1 Public Groups

In this usage mode HangOut allows large groups of people that share a common interest to collectively inform other users with similar interests on locations that are of interest to that group. These groups are open to anyone who is interested in joining the group. In this usage mode a mobile device user can discover an interest group, either from their cell phone or through a web interface that shows a list of available public groups and join the group without the need for any providing any identification information. HangOut provides a set of predefined common interest groups based on interest in music genre, food types for locating restaurants etc. The primary goal of public groups is to let people with similar interests collectively inform a user in the group where the most popular places are within user's vicinity.

2.2 Private Groups

Private Groups are similar to public groups in their expected usage model except that in order to participate in the group one needs authorization either from an existing member of that group or from the group administrator.

2.3 Personal Groups

In this usage mode HangOut allows a small group of people to share their precise location for a finite period of time. This usage model is useful in scenarios where a family or a group of friends want to track each other, such as a family visiting an amusement park or a group of people meeting for lunch. In this mode HangOut also allows participants to share a personal message along with their precise location information with members in their personal group.

2.4 Shared Landmarks

Orthogonal to the three usage models, HangOut also supports tagging places of interest that can then be shared with people in the same group as shared landmarks. Currently HangOut supports only text based shared landmarks where users can enter a descriptive text of a location from their mobile which will be displayed on the mobile device screens of other members of the group.

3. Location Representation in HangOut

Since HangOut is a location based service we first present a brief overview of a user's location is represented in our system. Location is encoded using a recursive quadrant representation as shown in Figure **1.** The GPS location of a user is first converted into a Mercator projection using the WSG84 world model (Datum)[1]. Mercator projects the world into a square planar surface. In our implementation the QRST representation is actually encoded as a three digit number sequence, named as X, Y, and Zoom. In this encoding the world is treated as a square grid of 2^{Zoom} X 2^{Zoom} squares. X and Y are the offsets from top left corner of the world. For instance, the highlighted quadrant in the figure marked as RST can be represented in binary format as X=110 and Y=011 and Zoom=3. A Zoom of 25 is assumed to be the maximum precision that location can be specified in. By default every GPS location is converted into a 25 bit X and Y values with Zoom set to 25. By using the quadrant representation the mobile device can efficiently decide the granularity it wants to use in sending its location update by simply changing the Zoom level.

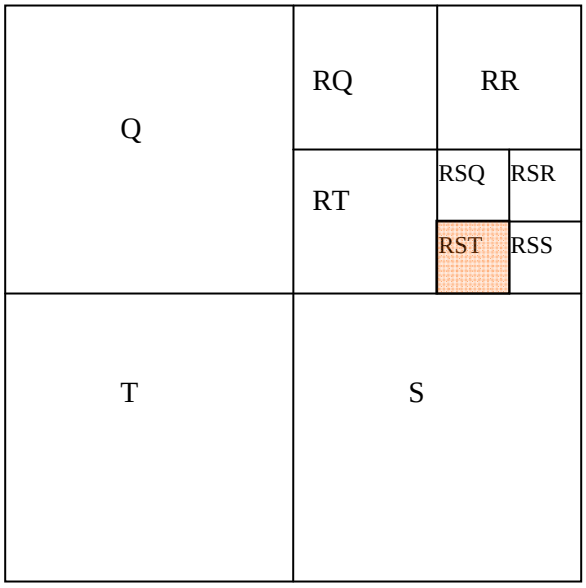

Figure 1 Quadrant Representation of World

4. HangOut Architecture

HangOut uses unique four-tier architecture to protect privacy and provide the functionality specified in the three usage models described in Section 2. In order to implement the functionality HangOut depends on the two fundamental operating functions, namely anonymous

updates and density requests. This section describes the components in the four tiers along with how the update and density requests are handled in HangOut.

4.1 Database Server

At the bottom of the hierarchy is a backend database server identified as SQL server in Figure **2.** The database server holds information related to all groups that are active in the HangOut system. For simplicity in our implementation we used one database table for each group. The database table for public and private groups holds the density (or popularity) information of each location. For instance, every entry in the public group table stores the location (X, Y, and Zoom) values followed by a cumulative number of times any mobile device has sent an update from that location. The database does not have any identification information of the mobile device that sent the update. Section 4.3 describes how the mobile device identification is stripped before sending the update to the database server.

For personal groups the database table holds one entry per each member of the group. The entry may contain any information that is relevant to that personal group in encrypted format. In our current implementation the entry contains precise location information (X,Y and Zoom=25) and any message that member of the group wants to share with other group members. The only information the backend server needs is the group number and member number. The group number is used by the server to identify which database table to access. The member number is then used to identify which entry within the selected table should be modified. The key for decrypting the personal group information is only available to the HangOut client devices that are members of that group. In essence, the information can not be decrypted by any intermediary (including the database server). The personal group information is as secure as the encryption key used by the group. Note that when a personal group member sends an update the database table entry for that member is overwritten with the updated information. Hence, there is no update track, even in encrypted format, available for an adversary. Since information is encrypted even if the adversary gets access to the database server and tracks all the updates there is no way to read the content as long as a strong key is used for encryption.

The backend server also supports density request functions from the mobile clients. A mobile client wanting to know the most popular locations for a given interest group initiates a density request. The requests typically contain the location which is specified as the top left corner of a quadrant that is of interest. The server accesses the database to find the cumulative count of updates to that location and sends that information back to the location server.

For personal groups the density request contains only the group number. The server then accesses the table associated with that group and sends the entire content of the table (information pertaining to every member of that group) back to the application server. For simplifying the message parsing the database server inserts the group number and the number of members (number of rows in the table) in the response message.

Apart from update and density requests the database also handles requests for creating new groups. The database server maintains a *groupinfo* table that holds the group number, description, group creation time, length of time the group is valid for, expected size of the group, and the group access key. The group access key is used to authenticate members of a group. Group creation requests specify a unique group number, group description and time limit. In addition, private and personal groups also specify a group access key. On receiving a group creation request the database server creates a new database table with the new group number. It then inserts a new entry into the groupinfo table that corresponds to the new group.

Apart from servicing client requests the database server also performs traditional housekeeping tasks. For instance, the database server scans the groupinfo table and using the creation time and time limit fields it determines the list of groups that are no longer valid and deletes database tables corresponding to those groups. It also creates indices on tables that are frequently accessed.

In our current implementation we use Derby [2] an open source database server for providing all the functionality. We used JDBC to interface the Derby server to the next tier in our hierarchy which is a Java Application Server called the Location Server.

4.2 Location Server

The layer on top of the database server is the location server which is the core of HangOut and provides all the functionality seen by the mobile device. The location server processes all the requests from the client device and appropriately routes them to the backend database server. The location server publishes an RSA public key which is used by the HangOut client to encrypt all communication with the location server. The encrypted communication prevents an adversary from viewing any information related to a request.

The location server first decrypts the incoming message to determine the request type. There are two primary requests serviced by the location server: update request and density request. For an update request the requestor (mobile client) specifies the list of groups that it wants to send an update for. The location server scans the list of groups in the request and sends an update request to the database server. The database server in turn cumulates the location density as described in **Section 4.1**.

The second request type is density request. These requests specify a group number and a location where the

client wants to know the density. The location is specified as the top left coordinates for the quadrant that the client is interested in. The location server first subdivides the request quadrant into a 32X32 grid of higher zoom level sub quadrants and issues a density request for each of the sub quadrants. By sub dividing clients requested quadrant into higher zoom quadrants we allow the client to requests density from a very coarse granularity, such as a 32KM X 32KM quadrant. The location server translates a single density request into a grid of 32 X 32 density requests and submits those requests to the database server. As mentioned earlier our database server sends the cumulative count of the number of updates seen at each of these quadrants. The location server then in turn locally aggregates the 32 X 32 grid of results and sends only a relative popularity matrix to the client. The relative popularity matrix is computed using the equation: $\log(density[i]/\sum_{j=1}^{1024} density[j])$. Simply stated, the location server first computes the sum of the 32 X 32 density data points returned by the database server. It then divides the density value of each sub quadrant by the total density of that area and finally takes a log of the divided value to generate the relative density.

The client can only see the relative density but not the absolute count of the updates. This prevents any adversary acting as a client from inferring update data. For instance, if the absolute values were given to the client the client can initiate multiple density requests in succession and compute the difference in absolute values to know if a new update has been sent between successive requests.

Just as is the case for the database server the location server does not have the identification information of the request originator. Hence, any adversary that gets access to the location server can only track request types but can not correlate the request type with a user.

Apart from the update and density requests there are two other types of requests processed by the location server. When a client wants to create a new group then client initiates a group creation request either from the mobile device using the HangOut interface or from a traditional web interface on a computer. The group creation request contains the group description, length of time that group is valid for, and expected size of the group (number of members). In addition for a private or personal group the client also sends a 128 bit group access key. The location server uses this information to generate a new group identification number. In our implementation we use a modified random number generator that guarantees uniqueness of the random number which is used as group id. The location server then requests the database server to create a group table and allocates appropriate resources, such as the disk space size, for that group based on group size. It also sends the access key to the database server which will insert the key and group description into a group information table. Note that the access key provided during the group creation will be used by location server to authenticate every update and density request for that group. In other words, every member of a private or personal group has to provide the right group access key for every update and density request. The location server sends the access key and group number to the database server which then verifies if the key matches the stored key. If there is no match the server simply drops the entire request without any further processing.

The location server is implemented using Sun Java System Application Server 9.1. The location server hides the implementation details of its functionality and provides a Java *webmethod* interface for each of the request types. The application server uses JDBC interface to communicate with the database server.

4.3 ID Proxy Server

The tier on top of the application server is the ID Proxy server. The identification proxy server is envisioned to be operated by an entity that is independent of the HangOut service provider. For instance, this server may be managed by the cell phone service operator. The mobile device communicates directly with the proxy server using HTTP post/get messages in our implementation. The HTTP message received by the proxy server from the client has two components. The first component contains the mobile device identification information, namely phone number, MAC address, cell tower id of the message origin. This component of the message is required for all cell phone communications as operator needs to appropriately charge for data communication costs. If the client uses a Wi-Fi channel for communicating with the proxy server only the MAC address of the Wi-Fi raedio is visible to the proxy server. The second component of the message contains information that is intended for the application server. The content of this second component varies based on the request type from the mobile device. The proxy server strips all the identification information from the message, namely the first component of the message, and passes on the second component of the message to the application server. Hence the location server does not have access to any identification information of the mobile device that is requesting for service. On the same note, as mentioned earlier in order to prevent the proxy server or any intermediary from viewing the request every request to the location server is encrypted using the public key of the location server.

4.4 Mobile Client

The top tier of our architecture is the end user, usually a mobile device client. The mobile device runs the HangOut client software. The client software provides the following functionality.

4.4.1 Mapping

HangOut client provides basic mapping capability. It reads the GPS location of the device either from the integrated GPS on N95 [3] or from an external Bluetooth enabled GPS receivers. The GPS coordinates are translated into X, Y, and Zoom level specification of a location. The client communicates with our in-house map tile server to fetch maps for that location. The client can use the keypad on the device to zoom in or out of an area. The map tile server uses a servlet based web server that receives a request for a map tile at a given X, Y, and Zoom level. It then provides a 256 X 256 pixel PNG formatted map of that location to the client. Note that our map tile server currently supports a maximum zoom level of 15. At the maximum zoom level each map tile roughly covers 32 X 32 meter area. The client prefetches 9 surrounding map tiles in order to provide a good user experience while panning the surrounding areas. In order to reduce the communication costs and delays the client caches the most recent *N* map tiles in the device flash memory. The size of the cache depends on the flash memory size on the device.

4.4.2 Group Management

Since groups are the foundation for HangOut usability, the client software provides extensive group management capabilities. Users can join an existing public group using a simple form based interface. In order to join the public groups users provide the group number and length of time they want to participate in that group. The group number can be obtained from a web or mobile device based group discovery tool. For private and personal groups users are required to provide the group access key. While joining personal group users are also required to provide a group private key, which will be described in **Section 4.4.3**. In addition a personal group user provides short identifier, such as the initials of the user's name, which will be used by other members of the group to easily identify the user.

Users can also create a new group. For group creation users may enter a brief description of the group, the time limit, and group size. For private group creation users are required to provide a 128 bit group access key. The client can auto generate the access key based on a combination of current time and MAC address of the device as a seed for a random number generator. The purpose of group access key, as described earlier in **Section 4.2,** is to allow the group users to authenticate themselves to the location server for requesting service.

HangOut client allows users the ability to send SMS invitations to their friends to join a group from within the application. For private groups SMS invitation contains the group id, group access key.

The group management module keeps maintains user's group membership information locally. For instance, this module keeps track of the time limit for a user's participation in each group. Once the time limit expires, the software removes the group from the list of user's active groups. Once a group is deleted the client does not send an update for that group. Similarly, client can not request density information for that group. By managing group memberships locally the software guarantees that no updates are seen by the server once the time limit expires.

4.4.3 Privacy in Personal Groups

One of the key capabilities of HangOut is to allow member of personal group to precisely locate each other and even exchange messages tagged with their location. In order to provide this functionality of locating individual members without compromising privacy HangOut takes a unique approach of using group private key only known to the group members. The group private key is either auto generated or can be entered by the users during the creation of the personal groups. The HangOut client stores the private key on the mobile device. In particular, the key is never sent as part of any communication with the proxy server. In order to join a personal group every member of the group needs to know the private key, in addition to the group access key. The use and distribution of private key is described here**.**

The group private key is a shared by all the devices that are members of a personal group. Every member of the group uses this private key to encrypt the location information and any personal message. In our current implementation we use a symmetric key (AES) encryption to encrypt the information. The encrypted information can be viewed only by other members of the group who have the same key. Every personal group request contains two parts. The first part contains the group number and member number. The second part contains the location and message information that is encrypted using the group private key. Note that all request messages – personal, private, public – are encrypted using the location server's public key.

Since HangOut allows users to send SMS invitations to join a group we used the SMS interface to exchange the group private key for personal group invitations. Since the current implementation uses SMS to exchange sensitive private key we recognize that an adversary who listens to the SMS messages may correlate the SMS information and future message requests and potentially identify user location.

5. Conclusions and Future Work

As mobile devices enter a new era with high speed connectivity and increasing compute capabilities a new class of applications called social networking applications is being showcased as the next revolution in mobile computing. In this class of applications each user in a social group contributes their knowledge about their

surrounding environments and the collective knowledge can then be exploited by the group members for a personal or social benefit. As the popularity of mobile social networks increases there is a growing realization that information collected about an individual user can compromise one's privacy and potentially security. It is in this context we developed HangOut a privacy preserving social networking application. Hangout protects user's private information not only from other malicious users but even from system administrators who may have unrestricted access to the backend server that is providing the social networking service. Hangout uses location and time distortions, symmetric key encryption where the keys are exchanged in a peer-to-peer fashion and several client side controls to aggressively protect privacy with minimal degradation in user's perceived service value.

6. References

[1] http://en.wikipedia.org/wiki/WGS84

[2] http://db.apache.org/derby/

[3] http://www.nseries.com/ces/#l=ces

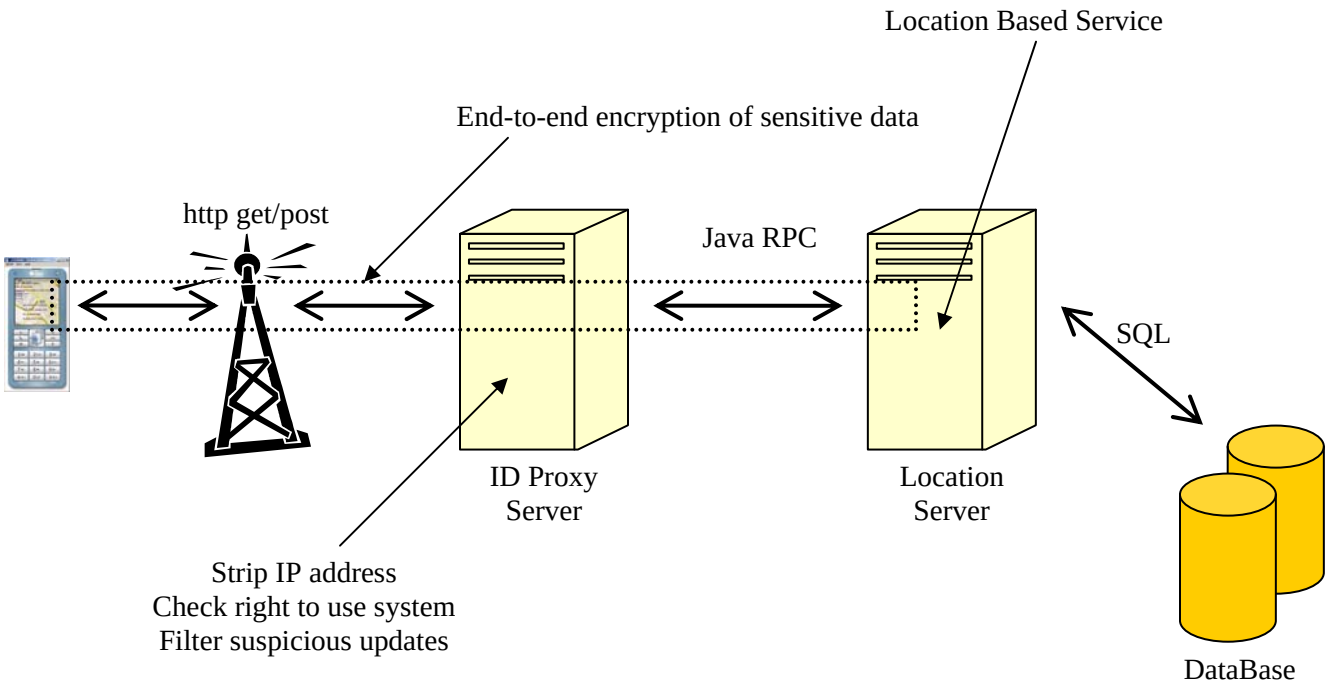

Figure 2 HangOut Four-Tier Architecture