A User Sensitive Privacy-Preserving Location Sharing System in Mobile Social Networks

Wei Cherng Cheng*, Masayoshi Aritsugi**

*Computer Science and Electrical Engineering, Graduate School of Science and Technology, Kumamoto University
2-39-1 Kurokami, Chuo-Ku, Kumamoto 860-8555, Japan

Abstract

Some types of social networking applications installed in the mobile devices require users to share own locations while using them. Adversaries glean users’ private information through shared locations via social networking application and public available geographic data. In this paper, we propose a user sensitive privacy-preserving location sharing system to avoid leaking. Users define sensitivity profiles to transform the public available geographic data into personal obfuscate region maps. Shared locations from obfuscate region maps provide close enough coordinates for application to use, but disconnected correlation between exact public available geographic data and user’s actual location to prevent malicious tracking.

© 2014 The Authors. Published by Elsevier B.V.
Peer-review under responsibility of KES International.

Keywords: Location Privacy; Obfuscate Region Maps; l-diversity

1. Introduction

Rapid development of wireless infrastructure and lightweight mobile devices, such as smart phones, tablets, and portable laptops, increase the internet utilization rate. Social networking application is one of utilizations being used widely now by mobile devices. Location information updated from mobile devices is fundamental intelligence for social networking applications. While runs the social networking application, it is aware of a user’s location, where
user visits as a result, our personal privacy data with sensitive contents and preferences are exposed together with the locations to malicious adversaries. However, we can’t deny the convenience brought by the social network applications which allow us to exchange the information quickly, and it is now the major tool for people to connect with others even though the applications are with potential risks.

According to the requisite of social networking applications for whether it is necessary to expose user location when use the applications, we categorize them into two types. Existing social networking applications, such as Facebook, Twitter and Line, accept users to switch off location sharing, and still able to use them [4]. We conclude the type as “passive location update” social networking application. In contrast, the newly merged on-line social networking applications, for example, Foursquare, Jelly, and Badoo transmit users’ location information as triggering requirement to utilize the service. The applications allow users to build a quick and personal social network by querying nearby strangers or friends via sharing own location. It needs periodic location update to keep all users staying in connection. We categorize this type as “active location update” social networking application. Personal information and privacy leaking risk increases particularly in the “active location update” type social network application because periodic location update from 3rd Party location-based server is needed. Thus, in this paper, we focus on the solution against to “active location update” type social networking application scenario, which users must share a related location with others.

To protect users’ location privacy, Damiani et al. [10] proposed a spatial obfuscating system based on user defined sensitive semantic locations model, that is, users’ location would not be exposed when they are in the sensitive location. However, adversaries can still track and retrieve users’ location with longer time monitoring since those semantic geographic data are available from 3rd party location-based service, such as Google, and Yahoo map. Wei et al. [5] used a cryptographic algorithm technique to anonymize user identity and location information in both cellular tower and social networking server. In practice, the existing cellular towers didn’t perform the pseudo dummies generation [2], encryption and decryption algorithm for certain mobile applications, and the approach suited to cellular networks only.

Hence, we propose a user sensitive privacy-preserving system which generates the location for sharing in mobile social network system. First, it transforms the publically available geographic data from 3rd party into a personal obfuscate region map in a separate server. It suits into heterogeneous network and keeps off the impacts from mobile devices’ hardware types, software version. Secondly, we introduce the sensitive threshold value along with l-diversity technology [6] to obfuscate the geographic data, thus, user concerned sensitive semantic locations can be safeguarded. The obfuscate region map is computed based on user defined threshold value and semantic location sensitivity value to transform the public geographic data into obfuscate regions, so users are cloaked in both sensitive and non-sensitive regions. Finally, the sensitive threshold value brings the obfuscate region map varying to prevent the adversaries gleaning users’ location privacy over time. The contribution of this work can be summarized as follows:

- The varied setting of sensitive threshold value, and semantic locations’ sensitivity value distinguishes one’s shared location from own self and others with same publically available geographic data.
- This paper presents safeguarded and transformed user location suits in social networking applications to share with nearby strangers.
- Flexible adjustment for user to update the obfuscate region map, and the map also generalizes users’ locations.

2. Related Work

On-line social networking applications request personal private information data pouring on to communication networks from mobile devices, ranges from phone model to personal details such as age, gender, and location [3,14]. Location privacy is broad range research topic, and various approaches have proposed to address the existing threats in current network environment. Location cryptographic [5, 9, 12] and location k-anonymity [6, 10, 11, 13, 16] are the two most popular location privacy preserving techniques. Wei et al. [5] proposed to encrypt user’s precise location before sending it to location-based server, and considered the location-based server as untrusted 3rd party, so generated and stored both precise and fake users’ IDs and locations in the cellular tower. Wen et al. [9] followed the same architecture but enhanced it with applying an extra symmetric key between cellular tower and user during
transmission. The proposal considered the cellular tower as a trusted 3rd party server, so it can perform fake users’ generation and users’ encryption. There are two concerns from the proposal. First, it limits connections with cellular tower only, and other network protocols, such as Wi-Fi, and AP don’t suit. Second, a cellular tower doesn’t perform as a functional server for specific mobile applications practically. Wernke et al. [12] proposed an arithmetic algorithm to break a user’s encrypted and obfuscated geographic location into n shares and stored them in n location-based servers. In architecture, the more location-based servers involved in the system guarantee better security, but it didn’t suit to “active location update” type social networking application, when fewer area location-based servers were involved.

In the location anonymity approaches, Xue et al. [6] and Bamba et al. [16] proposed l number of semantic type locations with k users in a region. The goal is to prevent user from other k-l others and take different semantic locations into account. They were adding the semantic locations as interference to obfuscate the users cloaking region, but lacking of varying flexibility to prevent the adversaries while adapting the same geographic data. The proposal from Damiani et al. [10, 11] extended the method with personal sensitivity value setting, but was not flexible for user to vary the sensitive and non-sensitive cloak region.

Our approach provides with solution which suited into different network protocols, less cost in architecture requests, and personally flexible adjustment for user sensitive request in protecting users’ locations while sharing them. It is also overcome the problem that users and adversary use the same geographic data base.

3. System Architecture and Design

We propose a user sensitive privacy-preserving system for mobile social networking, and it is composed by four fundamental elements. They are mobile devices, social networking service servers (here after, we use SNS server as abbreviation), public available 3rd party location-based service server (here after, we use LBS server as abbreviation), and obfuscate server. Mobile users use the on-line social networking application to search for others via SNS server. At the same time, the adversaries are able to use the same application to track and glean the target user’s privacy through user visited locations. In the proposal, we assume a trusted separated obfuscate server to transform the public geographic data provided by LBS server into obfuscate region map based on user profile. User’s actual location will be anonymized according to the obfuscated region map, and both actual and anonymized location store in the SNS server for querying purpose.

We promote a separated obfuscate server to generate the obfuscate region map and cloak users’ actual locations accordingly, because in-device software is restricted by couple disadvantages. For example, it consumes extra resources such as CPU, memory, and storage space, frequent location update computing drained off device battery, and consequently forced to update due to different mobile devices’ software (iOS, Android, and Windows) update [17]. In addition, obfuscate function in a separated server suits to heterogeneous network. Later on, it is easier to be integrated into SNS server provider as part of SNS network.

Fig. 1 shows the design of system architecture. User’s mobile devices get the coordinates from equipped GPS function, connected Wi-Fi, Access Point, or different cellular protocols. All users registered into the social networking application are requested to provide the user’s basic information and preference as profile to SNS server with location initially. User profiles are stored and maintained in the SNS server. SNS server, which receives the users’ cloak locations from the obfuscate server, also maintains the cloaked locations. We consider LBS server is untrusted and provides same geographic data publically to anyone. Adversaries can store the users’ trajectory with same geographic data through continuous and frequent location update via the social network application. The obfuscate server is responsible to generate an obfuscate region map from LBS server provided geographic data according to user profile.
After user completes the registration, SNS server sends a request with user coordinates to LBS server for providing a wider range geographic data to obfuscate server. The obfuscate server generates obfuscated region map according to user profile and LBS supplied geographic data. During querying process, it anonymizes users’ actual locations in the obfuscated regions into correlated regions’ coordinates. The cloak locations will then be sent to SNS server and be maintained there for user querying. User profile changes will update the obfuscate region map consequently. We detail obfuscating and querying processes in the subsequent section.

4. Obfuscate Region Map and Querying Processes

The obfuscate method in our proposal is transforming the geographic data from LBS server into obfuscate region map. Users’ actual locations are then cloaked, replaced with obfuscated region’s coordinates from the obfuscate region map. Obfuscating process is based on the user profile which includes the user specified sensitivity level for certain features in the area, corresponding threshold and query range. We first define the sensitivity model and user profile contents for obfuscate process. We then, explain the obfuscate process which adopts the Hilbert Curve movement process to conduct our personalized map. The sensitivity model includes the space definition and sensitivity criterion. A summary of notations used in this section is given in Table 1.

4.1. Space and Sensitivity Criterion Model

The concerned region in the LBS application refers to a two dimensional bounded area as reference space, and geometric objects in the reference space have spatial type compliant with geo-spatial standards. [1] In the proposal, we assume the obtained space geographic data from LBS as our reference space $\Omega$, and spatial types are described as feature types, denoting as $FTs$, while locations are described as features, denoting as $ft$. A region $r$, in the reference space $\Omega$ consists of different and disjoint features, such as road, sidewalk, parks, or varied constructions, $r \in \Omega$. Further features have an extent of region type.

Assume reference space $\Omega$ is a grid-based space subdivided into regular and sufficiently small unit size cells, denote as $c$ and $\Omega = \{C_1, C_2, \ldots, C_n\}$ while two cells $C_i$ and $C_{i+1}$ are adjacent cells sharing common boarder. [10] A region $r$ is a combination of boundary connected cells where $c \in r$.

For user’s location in unit cell $c$, we define the feature density function $D(ft,c)$ to quantitate the feature shares in the unit area. The feature density function is used to judge the extent of the unit cell. We define sensitivity level of feature types to be rated in value, and denote the sensitivity level as $v(ft)$ where $v(ft)$ has value between $[0, 1]$, and the greater value indicates users demand higher sensitive concerns when they are in the feature. For example, in a unit cell $c$ contains different features as a bank and a hospital in Fig. 2(a). The feature densities of $c$ with respect to bank and the hospital is calculated, and denotes as $D(\text{Bank}, c)$ and $D(\text{Hospital}, c)$. As shown in Fig. 2(a), within unit cell $c$, hospital density of cell $c$ is greater than bank density of cell $c$, i.e., $D(\text{Hospital}, c) > D(\text{Bank}, c)$.
We then take feature hospital’s sensitivity value \( v(\text{Hospital}) \) as sensitivity of unit cell \( c \) extent, denoted as \( E(\text{Hospital}, c) \), in Fig. 2(b). When a region \( r \) is made by combinations of different sensitivity of unit cells. We denote \( S(r) \) as the mean sensitivity value of a region to represent its sensitivity level.

4.2. User Profile

To avoid the adversaries glean the user’s real time location via publicly available geographic data, we use personal profile as our transformation variables to distinguish one from others. Personal profile defined as profile, includes a list of major concerned feature types \( FTs = \{ft_1, \ldots, ft_n\} \), for example restaurants, school, hospital or bank with respective sensitivity value \( V = \{v(ft_1), \ldots, v(ft_n)\} \). Sensitivity value 0 represents the location where user has no concerns about the feature type, and we exclude the case in our proposal. Sensitivity value 1 represents as extremely sensitive.

A sensitivity threshold, \( T \), needs to be decided by users as privatized basis to adjust the map. When feature type’s sensitivity is greater than the threshold value, it implies user’s actual location needs to be cloaked in the feature. The \( l \)-diversity technology means the \( l \) number of semantic type locations need to be cloaked. In our proposal, the \( l \) semantic types are feature types with sensitivity over threshold value. Users also need to define a querying range, \( Q_{min} \), for searching and sharing locations with others within the range. The user profile takes the form of the tuples: \(<FTs, V, T, Q_{min}>\).

Table 1. Summary of notations

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \Omega )</td>
<td>Reference space</td>
</tr>
<tr>
<td>( \Omega' )</td>
<td>Reference space with Obfuscate Regions</td>
</tr>
<tr>
<td>( ft )</td>
<td>features</td>
</tr>
<tr>
<td>( c )</td>
<td>sufficient small unit cell</td>
</tr>
<tr>
<td>( r )</td>
<td>combination of border connected unit cells</td>
</tr>
<tr>
<td>( D(ft,c) )</td>
<td>feature density function in unit cell</td>
</tr>
<tr>
<td>( v(ft) )</td>
<td>feature sensitivity level value</td>
</tr>
<tr>
<td>( E(ft,c) )</td>
<td>sensitivity of unit cell ( c ) extent with feature ( ft )</td>
</tr>
<tr>
<td>( S(r) )</td>
<td>mean sensitivity of region ( r )</td>
</tr>
<tr>
<td>( T )</td>
<td>Sensitivity threshold value defined by user</td>
</tr>
<tr>
<td>( Q_{min} )</td>
<td>Querying range surround user</td>
</tr>
</tbody>
</table>

4.3. Obfuscating Process with Hilbert Algorithm

Initially, a user provides location and profile to SNS server while starting the application. SNS server uses the initial coordinates to request LBS server for a wider range geographic data downloaded to obfuscate server. The
obfuscate server performs the transformation of geographic data into obfuscate region map, and also cloaks users’ actual location into correlated regions’ coordinates. The obfuscate region map in the server varies based on personal profile setting, so that each individual’s obfuscate region map will be different from others with same geographic data source. User’s location update is from SNS server with region location but not from LBS server. The obfuscate map regions update when users modify the profile set. The obfuscate process details as following.

Assume the downloaded reference space $\Omega$ is composed by unit cells as defined previously, $\Omega = \{c_1, c_2, \ldots, c_n\}$. Each cell follows the density function definition to grant the sensitivity value of unit cell corresponding with defined feature sensitivity value. Combined cells form a new region. Feature regions are located within space $\Omega$ can be arbitrary shape and size. The obfuscate process moves through all cells in $\Omega$ and merges sensitive cells with value higher than threshold value to become regions.

We choose the Hilbert Curve (also known as a Hilbert space-filling curve) to be our movement route as it compactly nevertheless fills all available cells in a fixed area. Similarly to the approach in [10], in the moving progresses process, Fig. 3, the unit cell sensitivity value is compared with user defined threshold value $T$. If the unit cell sensitivity value is higher than the threshold value, the cell is required to keep secrecy from others, so the cell merges with next cell in the movement route to enlarge the area and becomes the new region $r'$. The new mean sensitivity value for region $r'$ is calculated as $S(r')$. We compute and enlarge the new region by the same process until finally the mean sensitivity value is less than the threshold value.

If the unit cell sensitivity value is less than the threshold, we pass the cell and move on to next following cell and continue the process until we run through the whole cells in the space $\Omega$.

Algorithm: Fundamental Process for Region Map by Hilbert Curve Movement

1: function HilbertCurveMap(grid, profile, $T$) // Grid cells are using user profile’s feature sensitive value, $T$=threshold
2: RegionMap ← $\emptyset$ // Obfuscated Regions
3: $r$ ← $\emptyset$
4: $r'$ ← $\emptyset$
5: for $idx \leftarrow 0$ to maxHilbertIdx(grid) do // Run Hilbert Scan for grid in linear order from first cell till every cell has examined
6: cell ← getHilbertCell(idx); // Region initially starts as one cell
7: Add ($r$, cell)
8: if $S(r) > T$ then // $S(r)$ calculated as the mean sensitive value of region $r$
9: Add ($r'$, cell) // Enlarge Region $r'$ size till $S(r)$ value is less than $T$
10: else
11: Add (RegionMap, $r'$)
12: $r'$ ← $\emptyset$
13: $r$ ← $\emptyset$
14: end if
15: end for
16: Remap(RegionMap, profile) // Regions with shared boarder combine
17: return RegionMap
18: end function

Fig. 3. Fundamental process for region map by Hilbert Curve movement.

It is possible that initially user’s location cell is not cloaked if the cell sensitivity is less than threshold. To prevent this, we remap $\Omega$ by combining the cells with sensitive value under threshold but share the same boarder as a region. We now transform the reference space $\Omega$ with cells into the obfuscate region map $\Omega'$ with user’s sensitivity preference with regions.

Fig. 4(a) shows an example of $16\times16$ size obfuscated regions result. Users’ location will be cloaked and transformed as region’s location where they are. We take Fig. 3 as fundamental process. Furthermore, we remapped again for those sensitive regions with shared boarder by running the line 16 procedure again in Fig. 3 as revised process. Fig. 4(b) shows the result of the revised process. In the result of revised process, the obfuscated region number is less than the fundamental one, and have larger size of region, but further obfuscated.
4.4. Anonymization and Querying Processes

Instead of using dummies technique to achieve k-anonymity [2], we generalize all users’ actual locations with region coordinates in the obfuscate server. When a user, u, queries others in distance $Q_{min}$, SNS server filters users surrounded u within $Q_{min}$ range and forms a user list with locations denotes as $M(u)$. $M(u)$ is sent to obfuscate server for mapping all users’ actual locations into the regions’ coordinates based on user u’s obfuscate region map. The cloaked location users’ list denotes as $M'(u)$ and sends back to SNS server for user u’s querying.

The locations mapping and transformation in obfuscate server achieve two functions. First, user u’s actual location is cloaked into region coordinates, so user u’s sensitive location privacy is preserved. Second, user u’s location privacy is k-anonymously generalized as other k-1 users’ locations in the same region with user u are transformed to same region coordinates.

Fig. 5(a) shows an example, when user A queries with range $Q_{min}$, location list $M(A)$ is made in SNS server and there are n users in it. Based on user A’s obfuscate region map, the obfuscate server maps n users’ actual locations into 9 obfuscated region coordinates as shown in Fig. 5(b). The cloaked location users’ list $M'(A)$ is made of user IDs and region’s coordinates, and is sent back to SNS server for user A’s querying. In the same region where user A is, other k-1 users’ locations are transformed to same region coordinates. The cloaked location users’ list $M'(A)$ is location generalized.

5. Experiment and Evaluation

We divide the experiment into two parts for discussion. First, we show the obfuscate region map can safe guard user’s sharing location, and user concerned sensitive location can also be preserved with the system. Second, we demonstrate the flexibility of the region map by changing the threshold value user defined in the profile, and same LBS geographic source data will outcome different results to prevent malicious tracking.
5.1. Experiments setup

The experiment assumes user A initiates the social networking service in Traffic Center of Kumamoto City, Japan. The rectangular unit cell is defined as 10m×10m square size, and the city size is 1024×1024 grids, which is 10km×10km as the reference space. It is the regular size of a city. Total 50,000 users are randomly located within the reference space. The simulator obtains the road and street data set from OpenStreetMap data [15], which are assisted by commercial LBS companies such as Yahoo, or Google map. Spatial grid map fulfills with features and replaces with sensitivity value from user profile setting accordingly. In user A profile, we categorize all features types from geographic data fits into 14 types as in Table 2, and correlated sensitivity value given as a sample. We will adjust them as the experiment need. The experiment ran on a 64 bits Windows 7 desktop computer equipped with Intel Core i7-3770K 3.5GHz CPU and 16M RAM.

Table 2. Feature types and sensitivity values

<table>
<thead>
<tr>
<th>Item</th>
<th>Feature</th>
<th>Sensitivity Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Hospital, and Care Center</td>
<td>0.7</td>
</tr>
<tr>
<td>2</td>
<td>Bank, Post office</td>
<td>0.6</td>
</tr>
<tr>
<td>3</td>
<td>School</td>
<td>0.55</td>
</tr>
<tr>
<td>4</td>
<td>Traffic Station</td>
<td>0.5</td>
</tr>
<tr>
<td>5</td>
<td>Entertainment Location, Shopping stores</td>
<td>0.45</td>
</tr>
<tr>
<td>6</td>
<td>Accommodations</td>
<td>0.4</td>
</tr>
<tr>
<td>7</td>
<td>Department store and Mall</td>
<td>0.35</td>
</tr>
<tr>
<td>8</td>
<td>Dining restaurant</td>
<td>0.3</td>
</tr>
<tr>
<td>9</td>
<td>Religious Location</td>
<td>0.25</td>
</tr>
<tr>
<td>10</td>
<td>Public Facility (Parks, Tourism Locations)</td>
<td>0.2</td>
</tr>
<tr>
<td>11</td>
<td>Residence and Unclassified Building</td>
<td>0.1</td>
</tr>
<tr>
<td>12</td>
<td>Mountain or Unclassified Land</td>
<td>0.09</td>
</tr>
<tr>
<td>13</td>
<td>Others (such as Road, sidewalk, bridge)</td>
<td>0.09</td>
</tr>
<tr>
<td>14</td>
<td>Unreachable Location</td>
<td>1</td>
</tr>
</tbody>
</table>

5.2. Experiments and Result

Firstly, we show an example of varying obfuscate region map result from system. Fig. 6(a) gives an example of the 16×16 size original map from the source geographic data. It was transformed into obfuscate region map as in Fig. 6(b) with defined 14 features and with corresponding value in user profile from Table 2. The threshold value we use was 0.3. It means that user considers eight features types are sensitive (item 1 to item 8 in Table 2), and needs to be cloaked. 20 regions’ coordinates will be used for users. Fig. 6(c) shows different obfuscate region map shape when we update three tuples values (station, entertainment, restaurant) from (0.5, 0.45, 0.3) to (0.4, 0.2, 0.2) with the same threshold value. The outcome shows the regions reduced from 20 to 10, and now concerned sensitive feature types become six. Fig. 6(d) shows obfuscate region map which is made of user profile as in Table 2, and threshold value was 0.25. There are nine sensitive feature types, and regions amount reduce to 4. Either the features’ sensitivity or threshold value update will make the regions’ amount or size changed accordingly. For example, if a user is located in region 1 in Fig. 6(b), cloaked location will be different as in Fig. 6(c) and Fig. 6(d) when choosing right above corner as the region coordinates.
Fig. 6. (a) a 16×16 size original map; (b) region map with Table 2 feature sensitive value threshold=0.3 ;(c) region map with three features lower sensitive value ;(d) region map with Table 2 feature sensitive value threshold=0.25.

Secondly, in our system, because the obfuscate region map’s region coordinates are used to share with others, there is a tradeoff between the cloaked degree and privacy concern. Comparing the obfuscate results in the reference space (sensitivity feature and value in Table 2 with thresholds 0.2 and 0.4), the average region amount for threshold 0.2 in fundamental and revised processes shown as 4063 regions and 1178 regions; for threshold 0.4, 10963 regions, and 3244 regions correspondingly. In Fig. 7(a), we find out that 77% and 67% of the regions’ size are within 10 cells correspondingly in both processes for threshold 0.4, and only 52% and 58% for threshold 0.2. It means, 52% and above users’ location shift within 10 cells (1000 meter square) wide region, and it is suitable for user to safeguard the location and share it with others. Users who concern better privacy concerns can take lower threshold value, but also more chance to fall into the region with larger location deviation.

![Region size Analysis](image1)

![Region Amount vs Threshold Value](image2)

![Users vs region amount within Q min](image3)

Fig. 7 (a) Region size analysis with threshold; (b) Region amount variation in different threshold; (c) users in obfuscated regions.

Thirdly, Fig. 7(b) shows the variant threshold resulting in the regions change. In the circumstance that users sustain feature types’ sensitivity value, but update the threshold value, user concerned sensitive feature types varied. Lower threshold value implies greater amount of sensitive feature to be cloaked. Thus, the outcome of the region amount is less in the system. Comparatively, the revised process made region amount further less than fundamental process due to it bound the sensitive regions with shared boarder as a new region.

In our query process result, we evaluate the user A queried $Q_{min}$=1000m, and the outcome showed there are 3409 people (6.8% over total users) found within 123 regions and 76 regions for fundamental and revised processes respectively. Fig. 7(c) shows that users amount in those regions. There are 65 regions and 44 regions (52.8%, 57.8%) with 2 or more users from those two processes. User A is $k$-anonymized when he moves into the regions. It indicates our solution performs not only the location obfuscating, but also users’ identification generalization function.
6. Conclusion

Our user sensitive privacy-preserving location sharing system proposed the obfuscate server with algorithm. It is flexible to adjust the threshold value to update the obfuscate region map for users. It covers to sensitive l-diversity technique and provides a safeguarded location for users to share it in social networking application to find the nearby other. At the same time, user’s personal profile update consequently adjusts the obfuscate region map from public available map data to prevent threats of been gleaned. Separate server setup in our proposal suits to all different kind of network protocols, and consumes less effort and resource for mobile device users, furthermore, it will be more convenient for on-line social networking application provider to integrate the solution with existing process and provide a more secure environment.

References