J Inf Process Syst, Vol.10, No.4, pp.618~627, December 2014
http://dx.doi.org/10.3745/JIPS.01.0003

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

A Step towards User Privacy while Using Location-Based Services

Fizza Abbas* and Heekuck Oh*

Abstract—Nowadays mobile users are using a popular service called Location-Based Services (LBS). LBS is very helpful for a mobile user in finding various Point of Interests (POIs) in their vicinity. To get these services, users must provide their personal information, such as user identity or current location, which severely risks the location privacy of the user. Many researchers are developing schemes that enable a user to use these LBS services anonymously, but these approaches have some limitations (i.e., either the privacy prevention mechanism is weak or the cost of the solution is too much). As such, we are presenting a robust scheme for mobile users that allows them to use LBS anonymously. Our scheme involves a client side application that interacts with an untrusted LBS server to find the nearest POI for a service required by a user. The scheme is not only efficient in its approach, but is also very practical with respect to the computations that are done on a client's resource constrained device. With our scheme, not only can a client anonymously use LBS without any use of a trusted third party, but also a server's database is completely secure from the client. We performed experiments by developing and testing an Android-based client side smartphone application to support our argument.

Keywords—Location Based Services, Location Privacy, Point of Interests

1. Introduction

The latest trends in the world of smartphones that use sensing and positioning technologies have changed the ways in which people interact and communicate. Many location-sensitive services that determine a user's location with the help of GPS and forward the location information to the application have been introduced. Smartphone users search for their Point of Interest (POI), such as banks, hotels, etc., within their surroundings. These POIs are stored in a database controlled by the Location-Based Service (LBS) provider. LBSs provide information to the user

※ This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Ministry of Education, Science and Technology (No. 2012-R1A2A2A01046986).

※ This work was also supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MEST) (No. 2012-R1A1A2009152).

※ This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2014-H0301-14-1015) supervised by the NIPA (National IT Industry Promotion Agency).

※ This research was supported by the MSIP (Ministry of Science, ICT & Future Planning), Korea, under the ITRC (Information Technology Research Center) support program (NIPA-2014-H0301-14-1044) supervised by the NIPA (National ICT Industry Promotion Agency).

Manuscript received October 4, 2013; accepted October 3, 2014; onlinefirst November 27, 2014.

Corresponding Author: Heekuck Oh (hkoh@hanyang.ac.kr)

* Department of Computer Science and Engineering, Hanyang University Erica Campus, Ansan 426-791, Korea. (fizza_alvi85@yahoo.com, hkoh@hanyang.ac.kr)

according to the user’s query regarding a POI. LBS not only helps a smartphone user to find their area of interest, but also gives them information on traffic conditions, resources, vehicles, machines, emergency requests, mobile social networking, navigation systems, and finding friends at a particular location. As a result, the user reveals his/her current location via LBS and therefore this service may end up compromising his/her privacy. According to experts, the market growth for LBS has been increasing for the last four years. According to the Allied Business Intelligence Research Group, the expected revenue increase for LBS will be 92% by 2017. Among wireless subscribers, 332 million have used LBS applications in 2011 and industry experts expect that number to increase to nearly 2.2 billion by 2017 [1]. Today’s smartphones are more location aware. They can determine their exact location on their own, which has increased the demand for LBS.

Today’s market provides three deployment options for deploying LBS: (1) in-network, (2) hosted, and (3) hybrid. For in-network deployments, operators deploy LBS infrastructure and applications in their own data center. The smartphone service provider pays the capital expenditures for hardware and soft-ware, as well as for the operating costs associated with around-the-clock operation, maintenance, and monitoring. The major benefit associated with this option is having full control over the infrastructure of the smartphone service provider. For hosted deployments, location service providers deploy the LBS infrastructure in their data centers and provide the location services to the smartphone service providers. This option lets the smartphone service providers leverage the expertise of the location service provider to manage, maintain, and enhance the LBS. This allows the smartphone service provider to focus on its core competencies of subscriber acquisition and retention. This option has the additional advantage of removing a lot of the capital acquisition costs, as well as the operational and logistical burdens that are associated with hardware and software purchases, installation, scalability, maintenance, and around-the-clock operations. In hybrid deployments (also known as managed services), the LBS hardware and software are owned by the smartphone provider and are installed in either the smartphone service provider’s data center or in the location service provider’s data center. The location service provider manages them on behalf of the smartphone service provider. For any of these deployment options, the LBS applications may be deployed on the smartphone service provider’s servers, location service provider’s servers, or on an application service provider’s servers [2].

Privacy gives the user the autonomy to enter their personal information without any fear of it being stolen. Privacy is a right that is expected by user both individually and socially. This trust is very challenging to maintain between the user and LBS provider because LBS is conserved commercially and an adversary can easily compromise LBS or collaborate with it and can use the information against the user. LBSs have three main components: the user and mobile device, the provider of position technology (wireless network or mobile phone provider), and the service provider, as shown in Fig. 1. In this scenario the user location privacy is at risk and the service and network providers can easily track user activity. Similarly, an adversary who has access to this information can harm a user (i.e., theft or murder).

In this paper we propose a privacy-preserving scheme that involves computation from the client’s end during interaction with an LBS server. The server will not be able to know about the client’s location, nor will the client know about server’s database.

The remainder of this paper is organized as follows: in Section 2, our motivation for creating this scheme is presented. Section 3 shares information on related work that has already been carried out. In Section 4 we explain our proposed scheme and in Section 5 the conclusion and future work is provided.

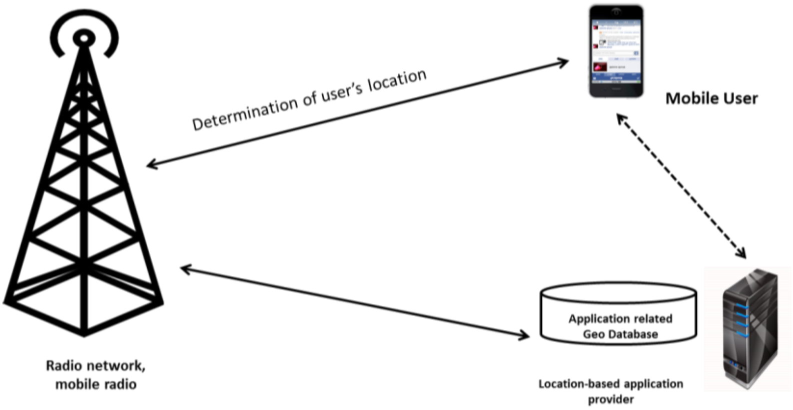

Fig. 1. Location-Based Service Infrastructure.

2. MOTIVATION

According to the *Merriam-Webster Dictionary*, privacy is defined as "The quality or state of being apart from company or observation" and "freedom from unauthorized intrusion (one's right to privacy)." Consider a scenario in which a hospital is interested in keeping the privacy of its patients, and intends to not leak any medical information that could be related to a specific patient, (e.g., AIDS patients). On the other hand, the hospital would like to promote the scientific research that is based (among other things) on the statistics of the information in the database. The hospital needs an access mechanism to the database that allows for certain 'statistical' queries to be answered, as long as they do not violate the privacy of any single patient.

Let's extend this discussion towards our agenda. In order to provide a user with any useful LBS service we need his location. Many users do not have a lot of concerns about their identity and/or location being revealed, but some of them are definitely concerned about these issues. Consider a scenario in which a user needs to find an AIDS clinic nearby. For obvious reasons, he/she needs the service without letting anyone know about his/her disease. If we only preserve their identity and reveal the location, then it is still possible to find the user's identity. Consider the scenario in which a user reveals his location to some LBS. An attacker can take the steps that are described below.

The attacker takes the location and finds it on Google Maps. Suppose that it is a suburban area and that it points to a home address. The attacker can easily use a telephone directory to find the person who resides there.

The challenge is to provide a privacy conscious user, with the required service, while still hiding his/her location. This becomes tricky because the LBS provider needs to know a user's location to provide him/her with any sort of LBSs. We need to find a way to preserve the privacy of a user, while still obtaining the required LBS.

3. Related Work

This section explains in detail the related work that has been done on this topic. Many authors have proposed approaches to preserve user privacy while using LBS. These approaches are divided into two categories: one is a trust-based approach, while the other is a trust free approach. Most of the approaches are based on trusted third parties (TTPs) that can be categorized as simple, policy-based, pseudonym-based, and anonymity-based. TTP-based schemes are well known in fields like e-Commerce. This is due to there being a solid trade-off between privacy, accuracy, and efficiency [3]. In a simple approach, a smartphone user sends a request to the LBS provider directly and depends on the honesty of LBS provider. A policy-based approach provides a framework in which users send queries to the LBS, where the LBS also sends a set of privacy policies that are known by the users and if these policies are not followed by the LBS, users can take legal action against the LBS provider in regards to the violation of privacy. In a pseudonym-based scheme, there is an intermediate party between the user and LBS provider. The pseudonymizer interchanges the real ID of the user with a fake one and forwards it to the LBS. However, the attacker can still acquire information about the actual user by linking the fake ID with his location. For example, let’s say that someone is resides in a palace and queries are continuously being sent from that palace, then an attacker can identify the query by its postal code [4]. The cloaking-based approach [5] is the more commonly used approach for preserving privacy and has been used by number of researchers. In a cloaking-based approach, which is usually called *k*-anonymity or spatial cloaking [6], a user’s location, along with the locations of a few of his/her neighbors are sent to a TTP that is known as the anonymizer. The anonymizer hides the user’s position among other users and sends a service query for all of them to the LBS provider. In order to do that, the anonymizer needs to be up-to-date with a pool of users. This scheme suffers from various drawbacks. First, the use of a TTP means that all of the users are trusting third party. The other problem is that if the TTP is compromised, then all of the other users’ locations are also compromised. Another drawback is that if a user asks for a service again, then he/she can be identified from among other *k* users. Cloaking or *k*-anonymity is efficient in processing due to the involvement of all of the calculations that take place on a TTP, but there is a high probability of identifying the user’s location [7].

The trust free approaches include private information retrieval (PIR) [8], cryptographic techniques like homomorphic encryption and the generation of dummies [9]. The generation of dummies [10] hides a user’s location and trajectory by sending several queries instead of only one. The drawback is a slower server response, which is due to the growing number of requests sent out by a user. Also, the LBS may suspect that it is under an attack and thus, the requests may be ignored. Once again, location privacy is dependent upon the number of queries sent by the smartphone user. Moreover, if this location information is exposed to the attacker, he/she can extract the true user’s information. With this approach, dummies must be intelligently selected; otherwise it can easily reveal information about the actual user. Cryptographic approaches [11] guarantee strong privacy at the cost of increasing the needs for processing. The techniques that are based on homomorphic encryption are efficient in preserving the user’s privacy because the LBS obtains zero knowledge from the encrypted query and sends back the required result to the client. The high computational and communication complexity makes this approach impractical. PIR [12,13] enables a user to submit a request to the database without revealing the actual request. This provides near-perfect privacy to the user, but suffers from computational costs or various assumptions about the limitations of the server's computational power. PIR protocols are expensive and require a significant amount of server resources.

In this paper we consider all of the limitations of previous approaches and present a novel hybrid approach that uses moderate computational resources and provides normal secure communication between the client and the LBS server without using a TTP.

4. PROPOSED SCHEME

In this section we discuss our working of proposed scheme, along with the implementation, results, and analysis of our scheme in detail.

For our scheme we used a client-side application that requests the server for all of the POIs for a given service in a particular region (such as a city, or depending of the population of POIs in that region, then, a portion of the city). The server finds all of the POIs in that region for the required service and sends the information to the client. The application then finds the closest coordinates from among those POIs and requests the server for the data for that particular POI. One of the potential threats to the user's anonymity is that if the POIs sent by server are located within specific location coordinates, then a clever server might be able to guess the approximate location of a user. For example, if there is a residential block and there is an ATM located nearby, then a server can guess that most likely the client is a resident of that residential block.

Keeping in mind the above-mentioned threat to the user's anonymity, we are proposing an approach that not only has improved efficiency while finding the nearest coordinates, but can also add fake coordinates in the scheme very efficiently. One interesting thing to note here is that the fake coordinates are listed among the actual POIs sent by the server. The scheme is based on the fact that the POIs are stored in the LBS database with their longitude and latitude coordinate values (hereafter, mentioned as *x* and *y* coordinate values), their name, address, etc. The LBS server then locates any POI with respect to its *x* and *y* coordinate values. In our scheme, the client-side application first sends a query to the server with the required POI service category name (e.g., an educational institute) and some region (e.g., a city). The server then finds all of the POIs for that service in that region from it database and sends it back to the client's application. Here the client's application performs two main tasks. After calculating the nearest POI and selecting K fake coordinates, the client-side application sends these to the server. The server can only guess the actual POI with a probability of 1/K. With two, K=2, and this becomes 0.33. Increasing the value of K can decrease the probability. After receiving the coordinates, the server searches for them in the database and returns the results to client. The client receives the coordinates and separates the required nearest coordinate with the help of its *x* and *y* coordinate values. It is worth mentioning that increasing the values of K does not affect the processing time of the client-side application. However, a larger number of Ks can result in the decreased performance of the server because of searching for extra (fake) coordinates. Hence, the value of K should be carefully selected based on how much anonymity a user wants and, accordingly, he/she has to wait for a server to respond. Fig. 2 shows overall working of proposed scheme.

4.1 Proposed Algorithm

Our proposed algorithm is as follow:

1. The client-side application encrypts and sends the service name and region to the LBS server.

2. The LBS server receives, decrypts, and then searches all of the POIs from a requested region for a required service. It then writes them to a text file, encrypts the file with its shared AES 128 secret key, and sends this back to the client's device.

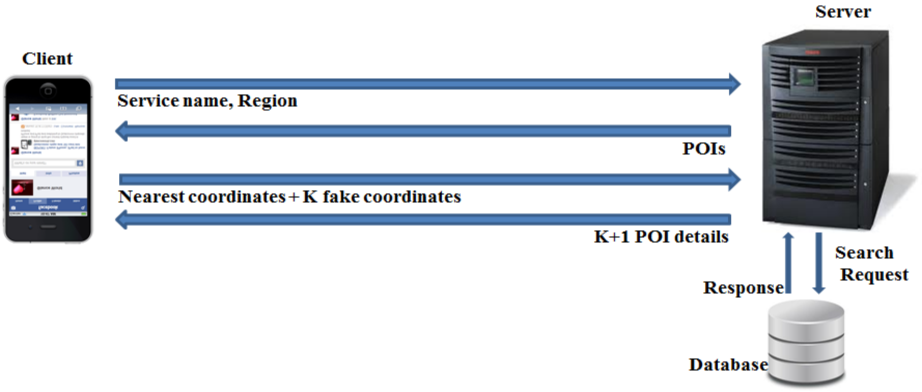

Fig. 2. Working of proposed scheme.

3. The client-side application receives the AES 128 encrypted file that contains the POIs from the server.
4. The client-side application decrypts the file with its shared secret key and stores the POI text file locally
5. The client-side application first calculates the distance from its current position coordinates, which it acquired with its GPS, with each of the POIs sent by the server and finds the nearest one.
6. The client-side application selects K random coordinates from the POIs sent by the server, while making sure that the randomly chosen K coordinates are different from the actual nearest coordinates.
7. The client-side application writes these coordinates into a text file, encrypts the file with its shared key, and sends this to the server.
8. The server receives K+1 POIs and is unable to guess which one is actually required by the client. It then finds the data (coordinates, name, address, etc.) of the K+1 POIs in its database, writes them back to a text file, and encrypts the file and sends it back to the client.
9. The client-side application receives the text file, decrypts it, and separates the data for the desired POI by matching its coordinates with client's device's coordinates acquired in step 5.

4.2 Implementation

We implemented this scheme with the help of a Java extension for Android in an Android-enabled Eclipse environment, known as the Android Developers Tools (ADTs). The concern with this implementation was to increase the speed of computations so that today's smartphone can search for the nearest coordinates and generate fake coordinates in a reasonable amount of time. The main computations in this scheme are to calculate the distance from the client's location coordinates with all of the coordinates of the POIs one by one and to then sort them so that we could find the closest one. After this, our application selects the K coordinate randomly from the POIs sent by the server, while making sure that the chosen fake coordinates are different from the found nearest coordinates. The pseudo code for the client application is given below.

```
READ Xdevice,Ydevice from the smartphone GPS device
READ Xcoordinate_array, Ycoordinate_array[] from “server_textfile.txt”
i=0,x=0,y=0,current=0,previous=1000,Xtarget=0,Ytarget=0,Xfake1=0,Yfake1=0,
Xfake2=0,Yfake2=0;
rand1=random(integer_range);rand2=random (interger_range);//random indices generation
REPEAT
x = Xcoordinate_array[i]
y = Ycoordinate_array[i]
IF rand1==i
Xfake1 = Xcoordinate_array[i]
Yfake1 = Ycoordinate_array[i]
END IF
IF rand2==i
Xfake2 = Xcoordinate_array[i]
Yfake2 = Ycoordinate_array[i]
END IF
dist[i] = SQRT((x-Xdevice)*(x-Xdevice)+ (y-Ydevice)*(y-Ydevice))
current = dist[i]
IF current < previous //comparison for sorting
Xtarget = x
Ytarget = y
previous = current
ENDIF
i++
UNTIL EOF
WHILE (Xtarget==Xfake1 && Ytarget==Yfake1)
//making sure that the fake coordinates are different from the actual nearest coordinates
            rand1=random(integer_range)
//generating the random index again
          Xfake1=Xcoordinate_array[rand1]
          Yfake1=Ycoordinate_array[rand1]
END WHILE
WHILE (Xtarget==Yfake2 && Ytarget==Yfake2) //making sure that the fake coordinate is different from the actual nearest coordinates
            rand2=random(integer_range)
//generating the random sequence again
          Xfake2=Xcoordinate_array[rand2]
          Yfake2=Ycoordinate_array[rand2]
END WHILE
WRITE “Nearest coordinates=”,Xtarget,Ytarget
WRITE “Nearest coordinates=”,Xfake1,Yfake1
WRITE “Nearest coordinates=”,Xfake2,Yfake2
```

4.3 Results

Our test bench was comprised of a Galaxy Note 2 smartphone. The specifications for this smartphone are as listed below.

Processor: quad-core 1.6 GHz Cortex-A9
RAM: 2 GB
Operating System: Android OS, v4.1.x Jelly Bean

The results were generated after installing the Android executable file on the client's mobile. The main emphasis was on two parameters. The first parameter in our application is the time taken by the client-side application to calculate the nearest coordinates from a different number of POIs sent by the server. The other parameter is the value of K (fake coordinates), where we tested our application with K=2, K=3, and K=4 (2, 3, and 4 coordinates, respectively).

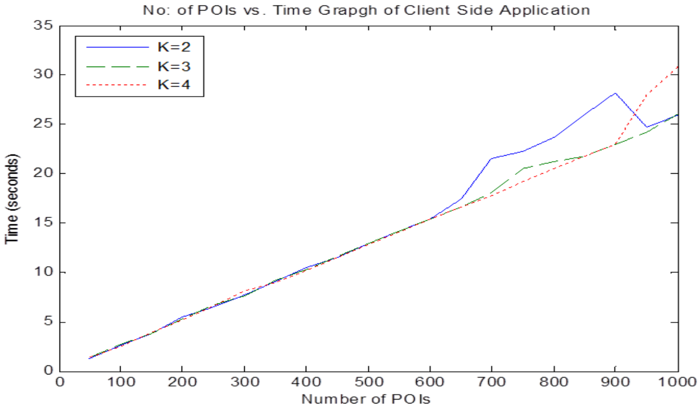

Fig. 3. Total time taken by client-side application for computing the coordinates for nearest POI and generation of appropriate K fake coordinates.

The results of our client-side application testing are shown in Fig. 3. For up to almost 650 POIs, there is no difference in the execution time for increasing the values of K. After 650 there is little difference, which is probably due to the extensive calculations while computing distances, sorting these distances to get the nearest one and in the generation of appropriate fake coordinates from out of more than 700 POIs. Here, our application responds in only 12 seconds while processing 500 POIs for all Ks. Similarly, it processes 1,000 POI's in under 30 seconds.

It is interesting to note that the given numbers of POIs in a city are fairly limited. For example, hospitals, train stations, or restaurants do not range in the thousands.

The results in Fig. 3 show that a large number of categories can be searched in seconds. The linear graph also indicates that the region can be extend by analyzing what is the tolerable execution time limit a user can wait to ensure that his/her privacy is maintained, while still obtaining the required service anonymously.

4.4 Analysis of the Scheme

In this scheme we have not used a TTP, therefore, our scheme does not require any TTP involvement, nor can it lead to other users' locations being compromised. The communication between the client and server is secured by AES-128 symmetric encryption. The use of fake coordinates guarantees that a server cannot guess a client's location because it does not get a single POI but instead receives K+1 POI data requests. By doing so, the probability of guessing the nearest coordinates is 1/K. For increasing the values of K, our application works well, which is very practical. Additionally, the server's data privacy is guaranteed as the client only gets a list of POI coordinates and nothing about the POI's identity. Moreover, the results show that our application is finding the desired results fairly quickly, which demonstrates that it is a practical approach. By executing and taking results on a smartphone we show that our scheme can be implemented on smartphones.

One of the significant features of this scheme is its simplicity. With normal database queries a server finds the required POIs, writes them into text files, and applies AES encryption. All of these steps are easy to implement. Our scheme does not require any special hardware or software for implementation on the client's or server's side of things. Another feature of this scheme is that if this client-server communication is ever compromised, the attacker will only get a location coordinates text file sent from the server to the client or a location that is sent to the server by the client. Due to the presence of fake coordinates, the attacker will face the same issue of guessing as with server. Similarly, the attacker cannot learn anything about the server's database as well.

5. Conclusion and Future Work

In this paper we focused on the location privacy of smartphones users. We have given a robust scheme for using LBS anonymously. This scheme is both efficient and practical. We have tested the client-side application in our scheme on a current smartphone and found that it can be implemented in a practical manner In the future, we aim to make our client-side application faster so that it can do computations in less time on a mobile with execution capabilities that are more limited than that for a smartphone. Moreover, we aim to develop a Java enabled version for this application side-by-side with the Android one.

References

[1] Allied Business Intelligence Research Group, "Location based services," https://www.abiresearch.com/market-research/service/location-enabled-services/.

[2] W. J. Buchanan, Z. Kwecka and E. Ekonomou, "A privacy preserving method using privacy enhancing techniques for location based services," *Mobile Networks and Applications,* vol. 18, no. 5, pp. 728-737, 2013.

[3] P. Kotzanikolaou, E. Magkos, N. Petrakos, C. Douligeris, and V. Chrissikopoulos, "Fair anonymous authentication for location based services," in *Data Privacy Management and Autonomous Spontaneous Security.* Heidelberg: Springer, 2013, pp. 1-14.

[4] E. Snekkenes, "Concepts for personal location privacy policies," in *Proceedings of the 3rd ACM Conference on Electronic Commerce*, Tampa, FL, 2001, pp. 48-57.

[5] S. Wang and X. S. Wang, "In-device spatial cloaking for mobile user privacy assisted by the cloud," in *Proceedings of the 11th International Conference on Mobile Data Management (MDM2010),* Kansas City, MO, 2010, pp. 381-386.

[6] C. Y. Chow, M. F. Mokbel, and X. Liu, "A peer-to-peer spatial cloaking algorithm for anonymous location-based service," in *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems (GIS2006)*, 2006, pp. 171-178.

[7] G. Zhong, I. Goldberg, and U. Hengartner, “Louis, lester and pierre: three protocols for location privacy,” in *Privacy Enhancing Technologies*. Heidelberg: Springer, 2007, pp. 62-76.

[8] G. Ghinita, P. Kalnis, and S. Skiadopoulos, “PRIVE: anonymous location-based queries in distributed mobile systems,” in *Proceedings of the 16th International Conference on World Wide Web*, Alberta, Canada, 2007, pp. 371-380.

[9] A. Solanas, J. Domingo-Ferrer, and A. Martínez-Ballesté, “Location privacy in location-based services: beyond TTP-based schemes,” in *Proceedings of the 1st International Workshop on Privacy in Location-Based Applications (PiLBA2008)*, Malaga, Spain, 2008, pp. 12-23.

[10] A. Pingley, N. Zhang, X. Fu, H. A. Choi, S. S. Subramaniam, and W. Zhao, “Protection of query privacy for continuous location based services,” in *Proceedings of the 30th IEEE International Conference on Computer Communications (INFOCOM 2011)*, Shanghai, China, 2011, pp. 1710-1718.

[11] Y. Gahi, M. Guennoun, Z. Guennoun, and K. El-Khatib, “Privacy preserving scheme for location-based services,” Journal of Information Security, vol. 3, no. 2, 2012, pp. 105-112.

[12] F. Olumofin, P. K. Tysowski, I. Goldberg, and U. Hengartner, “Achieving efficient query privacy for location based services,” in *Proceedings of the 10th International Symposium of Privacy Enhancing Technologies (PETS2010)*, Berlin, Germany, 2010, pp. 93-110.

[13] A. Khoshgozaran and C. Shahabi, “Private information retrieval techniques for enabling location privacy in location-based services,” in *Privacy in Location-Based Applications*. Heidelberg: Springer, 2009, pp. 59-83.

Fizza Abbas

She received her Bachelor’s degree in Computer System Engineering from Quaid-e-Awam University of Engineering, Science and Technology (QUEST), Pakistan in 2007. She received her Master’s in Communication System and Networks from Mehran University, Pakistan in 2011. Currently she is pursuing her Ph.D. in Computer Engineering from Hanyang University, South Korea. Her research interests are Security and Privacy in social network services, mobile social networks, cloud computing, mobile cloud computing and vehicle ad hoc networks (VANETs). She has six years of teaching experience and was working as Assistant Prof. in QUEST Pakistan before taking study leave and coming to South Korea.

Heekuck Oh

He received his B.S. degree in Electronics Engineering from Hanyang University in 1983. He received his M.S. and Ph.D. degrees in Computer Science from Iowa State University in 1989 and 1992, respectively. In 1994, he joined the faculty of the Department of Computer Science and Engineering, Hanyang University, ERICA campus, where he is currently a professor. His current research interests include network security and cryptography. Prof. Oh is the senior executive vice president of Korea Institute of Information Security & Cryptology, and is a member of Advisory Committee of Digital Investigation in Supreme Prosecutors' Office of the Republic of Korea. He is the corresponding author of this paper.