Providing Location-privacy in Opportunistic Mobile Social Network

by

Rui Huang

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the MASTER OF APPLIED SCIENCE degree in
Electrical and Computer Engineering

Ottawa-Carleton Institute of Electrical and Computer Engineering
School of Electrical Engineering and Computer Science
University of Ottawa

Abstract

Users face location-privacy risks when accessing Location-Based Services (LBSs) in an Opportunistic Mobile Social Networks (OMSNs). In order to protect the original requester's identity and location, we propose two location privacy obfuscation protocols utilizing social ties between users.

The first one is called Multi-Hop Location-Privacy Protection (MHLPP) protocol. To increase chances of completing obfuscation operations, users detect and make contacts with one-hop or multi-hop neighbor friends in social networks. Encrypted obfuscation queries avoid users learning important information especially the original requester's identity and location except for trusted users. Simulation results show that our protocol can give a higher query success ratio compared to its existing counterpart.

Another protocol is called Appointment Card protocol (ACP). To facilitate the obfuscation operations of queries, we introduce messages called the Appointment Card (AC). The original requesters can send their queries to the LBS directly using the information in the AC so that the query time cost of ACP is similar as no-privacy protocols ensuring that the original requester is not detected by the LBS. Also, a path for reply message is built when the query is sent, thus saving lots of time in replying queries. Simulation results show that our protocol can reach a higher query success ratio comparing to the existing protocol.

We also create a new OMSN simulator, called OMSN Routing Simulator (ORS), involving social network to provide a better evaluation of OMSN protocols. It shows more efficient and reliable performance in our experiments.

Acknowledgements

First, I would like to express my sincere gratitude to Prof. Amiya Nayak for his continuous support, motivation, advice and immense knowledge throughout my thesis work. I could not have imagined having a better mentor for my Master's study.

Also, I would like to thank my family and friends, especially Yichao Lin, for providing me with unfailing support and encouragement throughout the process of researching and writing my thesis. This accomplishment would not have been possible without them. Thank you.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Location-privacy is becoming a major concern in the Opportunistic Mobile Social Network (OMSN) which is a kind of Delay Tolerant Networks (DTNs) [3] featuring lack of continuous connectivity. More specifically, in OMSNs, it is not necessary for senders to have an end-to-end routing path to their destinations. Users make contacts when they encounter each other. LBSs are common applications in OMSNs and they are widely used in "military and government industries, emergency services and the commercial sector" [13], especially after the proliferation of localization technologies, like GPS. Many people access to LBSs with their portable devices and send their location to LBS providers, which expose their trace in front of the LBS. In this case, LBS users face a continuous risk that their location may be leaked from LBS applications. That makes people unwilling to use LBSs. Thus, protect location privacy has been a critical issue in LBSs.

1.0.1 Motivation

Location-Based Services (LBSs) which use the location information of users can be considered as ?two types of application design: push and pull services? [13]. For example, you may receive an advertisement when you enter an area, which is a push service; if you look for the nearest restaurant, you must pull information from the network. It is obvious that you must expose a location if you use LBS. When you use a LBS application to find the nearest restaurant, you actually tell the LBS provider your location and your next destination which might be a restaurant nearby. If attackers can access the LBS provider's database, they can learn that information. Then you may receive a lot of advertisements from surrounding restaurants.

The above inference attack just bothers you with advertisements, but it becomes more dangerous when you send more queries to the LBS provider. If you use the former LBS application 10 times in a day, attackers can learn your trace from the information so that they can infer more information including your identity and home address. ?Hoh used a database of week-long GPS traces from 239 drivers in the Detroit, MI area. Examining a subset of 65 drivers, their home-finding algorithm was able to find plausible home locations of about 85%, although the authors did not know the actual locations of the drivers' homes? [6]. Therefore, the LBS provider is considered as a semi-trusted party so that users must protect their location-privacy when they use LBS applications.

The OMSN is defined "as decentralized opportunistic communication networks formed among human carried mobile devices that take advantage of mobility and social networks to create new opportunities for exchanging information and mobile ad hoc social networking" [11]. In other words, OMSN is a kind of mobile ad hoc network, and the information and technology of social network are also used in it. People carrying smartphones which contains WiFi or Bluetooth equipments can compose a typical OMSN. Since the WiFi, Bluetooth equipment on smartphones can be used for discovering other devices and direct communication between devices, users can communicate with others who are in their communication ranges without any infrastructure, which is the basic requirement of an OMSN. Since users in OMSN could also use LBS applications, their location-privacy must be protected. Besides, the information in the social network allows users to identify friends. As a result, we can design a location-privacy protection protocol using the social network.

1.0.2 Objectives

Disconnection, decentralization and highly delays are obvious features for OMSN so that a decentralized strategy may benefit for the location-privacy protection protocol in OMSN. Besides, decreasing interactions between users and servers can significantly cut down the time cost. To prevent attackers from learning users' private information, the protocol should obfuscate users' location and hide their identities.

1.0.3 Contribution

We propose two new decentralized location-privacy protecting protocols for OMSN so that they do not need a three-party server which is used to obfuscate queries for users. We also create a new simulator for OMSNs called OMSN Routing protocols Simulator (ORS) to evaluate our protocols.

The first protocol which we propose is a distributed location-privacy algorithm, called Multi-Hop Location-Privacy Protection (MHLPP). It guarantees location-privacy and achieves a higher query success ratio. The introduction of social networks enables us to hide the original requester's information behind his friends. When a user wants to send a query, he starts to look for friends based on information in his social network. He sends his query to the first encountered friend who is then responsible for forwarding the query to the intended location. This friend can also pass the query to one of his friends when they encounter. When the distance between the user carrying this query and the original requester exceeds a specified threshold, the user sends the query to the LBS server directly without having to find a friend to pass on. At that time, he also replaces the original requester's information with its own identity and location, which enables the LBS server to receive the query without any information about the original requester. After receiving the query, the LBS server replies to the last friend (the user sending the query to the LBS) who then transmits it to the original requester.

Our second protocol is also a distributed location-privacy algorithm called Appointment Card Protocol (ACP), which aims to guarantee location-privacy and reach a higher query success ratio. The introduction of social networks enables us to hide the original requester's information behind his friends. We introduce the Appointment Card (AC) as a kind of intermediary which records a serial of agents. The original requester sends his query using the identity of the first agent in the AC to the LBS, which prevent the LBS learns the identity of the requester. The reply of the query can be delivered back to the original requester along with the same serial of agents as in the AC one by one. The last agent called the trusted agent is a user in the social network of the original requester, in other words, he is a friend (or a friend of friends) of the original requester. The trusted agent separates the stranger-agents and the original requester so that no stranger knows the identity of the original requester. The query-delivery success ratio of the ACP is in a similar level as the no-privacy protocol comparison in our experiment.

Our new simulator ORS is inspired by a well-known simulator the ONE [5], which is used by a lot of researchers to test their models and protocols in DTN. The major reason why we create a new simulator is that the ONE is not designed for OMSN so that there is no social network context in it. Adding social network information into the ONE simulator must modify the basic structure of it, which might import risks for the correctness of our experiments. We also use more reliable and efficient algorithm in our simulator, so that our simulator is about 2 times faster than the ONE in the case where we test our protocols.

1.0.4 Thesis Organization

This thesis is organized as follows:

Chapter 2 describes the concept of Mobile Ad hoc networks, Delay Tolerant Network, Location-Based Services and social networks. We give an overview of the existing location-privacy protocols.

Chapter 3 describes our first protocol MHLPP, which is explained in detail through architectural and mathematical instances. The comparison is made with a similar location-privacy protocol.

Chapter 4 elaborates on the second protocol ACP. We introduce a description of how it works and how it enhances the location-privacy. We also use an example to show the whole process.

Chapter 5 shows our new simulator. Also, we introduce the algorithms which are used in the simulator.

Chapter 2

Background and Related Work

2.1 Mobile Ad hoc networks

"Ad-hoc networks are infrastructure-less and cooperation-based networks which means that the network topologies must be decided by the network sensors themselves" [19]. In other words, the major feature of ad-hoc networks is that they have no infrastructure.

The Mobile Ad hoc network (MANET), also called the Wireless Ad-hoc Network (WANET), is a decentralized type of wireless network, and there is no pre-existing infrastructure managing the network, neither. In the MANET, because users move independently, the topology change frequently. Without the help of infrastructure, users can only communicate when they encounter each other. In other words, users communicate only if they are covered by each other's communication range. Because of the previous features, the MANET can be viewed as a kind of delay-tolerant network (DTN) which lacks continuous network connectivity. Then MANET can hardly establish instantaneous end-to-end paths, which compel the routing protocols in MANET to use a characteristic strategy "store-and-forward". Uses forward messages when they encounter others, or they just store messages, as shown in Figure 2.1. The user A has a message to the user C, so he sends the message to the user B when they encounter each other. The user B stores the message and moves, until he meets the user C. When the user B and the user C make contact, the user B sends the message to the user C, so that the message is delivered. It is obvious that whether the message can be delivered depends on the movement of users. If the user B never meet the user A or the user C, the message must be dropped after the timeout. The delivery process might be low success ratio and time-consuming.

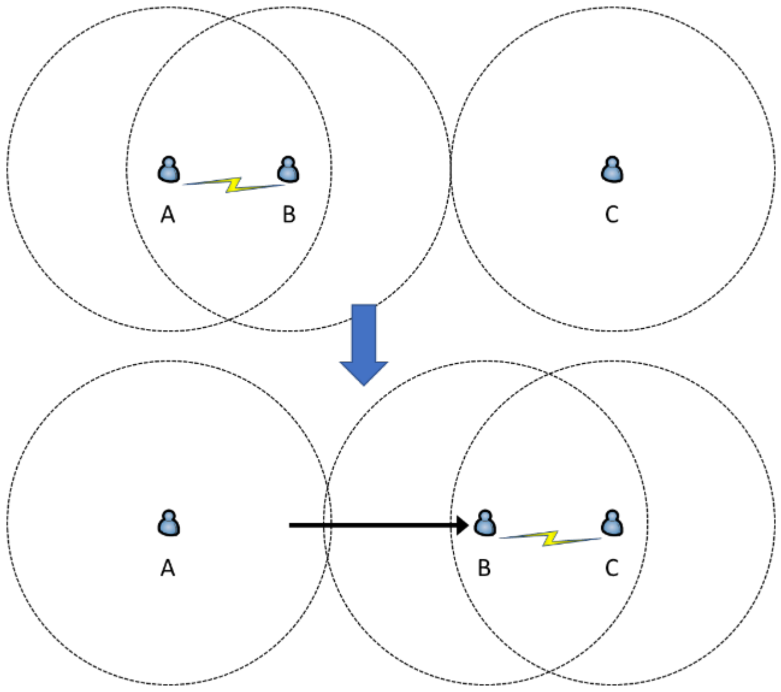
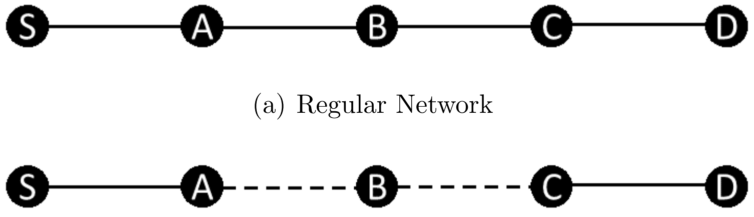


Figure 2.1: MANET

2.2 Delay Tolerant Network

Regular networks, e.g. Internet, always have end-to-end paths. As shown in Figure 2.2, the node S and the node D are connected by A, B, and C. Their maximum round-trip time is not excessive, and their drop probability is also small. Comparing to the regular network, there is a class of challenged networks [3] which lacks end-to-end path and suffering from high latency and long queuing times. As shown in Figure 2.3, when the node B is not working, there is no connection between the node A and the node C, so that extensive latency is inevitable if the source S sends a message to the destination D.



(a) Regular Network

(b) DTN

Figure 2.2: The comparison between regular networks and DTNs

Since the challenged networks have significant differences with the regular networks, researchers introduce a new architecture Delay Tolerant Network (DTN) to deal with the unique features in the challenged works. Terrestrial Mobile Networks, Exotic Media Networks, Military Ad-Hoc Networks and Sensor/Actuator Networks are typical DTNs.

2.2.1 Spray and Wait

The Spray and Wait is a known protocol for DTNs. Although it is not as efficient as protocols on internet, it works in DTNs. A message is initialized with several copies, a part of which are given to others when users encounter each other. Users keep forwarding copies until they each have only one copy of the message. When a user carrying at least one copy of the message encounters the destination, he gives all his copies to the destination to finish the delivery. The Binary Spray and Wait (BSW) is an optimized version of the Spray and Wait, which is also used in our protocols. The user who creates the message also makes several copies of that message. He gives half of the copies the user whom he encounters so that both he and the other user have a half of the copies. The users who get copies of the message continue to give a half of copies holding by them to others until they have only one copy. The remained one copy is given to the destination only.

2.2.2 Other Protocols

2.2.2.1 Direct Contact Scheme

The simplest strategy for DTN is that the source holds the message until he meets the destination, which is called the direct contact scheme. So, a direct path between the source and the destination is necessary for a success delivery. It is possible that the source never comes in contact with the destination so that message is not delivered.

2.2.2.2 Replica Based Protocols

The replica based protocols work by making several replicas of the message so that users can retransmit them upon connection establishment. The former BSW is also a kind of replica based protocols. Comparing to the direct contact scheme, the replica based protocols make it more possible for messages to be delivered. But they require more resources than the direct contact scheme because they need more memory to store the replicas. Therefore, making a reasonable decision for the message replication and dropping is the key to these kinds of protocols.

2.2.2.3 Knowledge-Based Protocols

Different from the former replica protocols which require no knowledge about the topology of the network, the users in knowledge-based protocols try to evaluate their own view of the topology of the network so that they can make better forwarding decisions. However, the topology changes so frequently that it is hard for users to have an accurate topology.

2.2.2.4 Coding Based Protocols

Authors in [7] and [1] make the approach by introducing coding techniques into routing protocols. Instead of making a few replicas, coding based protocols encrypts data to make a large number of message blocks. If the destination receives a part of the blocks, he can decrypt the message.

2.3 Location-Based Services

The Location-Based Services (LBS) use users' location information to provide services, which is shown in Figure 2.4. Your smartphone can detect your coordinate and send it to LBS application server which is responsible to provide service based on the coordinate. The point of interest and the location advertisement are familiar instances of LBS. The LBS providers collect users' location information to provide services, which makes LBS providers a significant target of attackers. When attackers access the databases of the LBS providers, they can learn all sensitive information in the servers. So, the LBS providers should be viewed as semi-trusted, which might expose information in front of vicious attackers.

Users face risks of information breach when they access a semi-trusted LBS provider because anyone who has access to data in LBSs is able to steal and misuse LBS users' location-privacy. Considering that LBSs rely on location-aware computing, it is unavoidable to leak users' location from LBSs. Therefore, balancing ?these two competing aims of location privacy and location awareness? [2] is always a challenge.

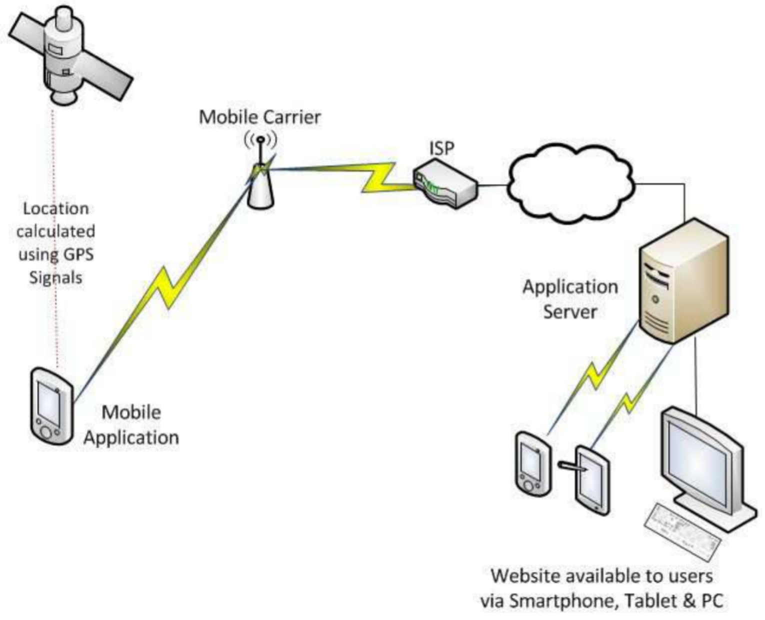

Mobile Carrier
ISP
Location calculated using GPS Signals
Application Server
Mobile Application
Website available to users via Smartphone, Tablet & PC

Figure 2.3: LBS [12]

2.4 Social Network

Social networks which contains social interactions and personal relationships are used in location-privacy protection so that users can determine who is trustful. In [15], the relationship of a pair of people (e.g., a user i and a user j) is described as a pair of directed relationship strengths. The directed relationship strength from the user i to the user j can be denoted by RS^{ij}. We should notice that RS^{ij} might be unequal to RS^{ji}, because the user i might like the user j so much while the user j just views the user i as an acquaintance. Based on the relationship strength, researcher can estimate the relational tie between two people and predict whether they are friends or strangers.

The relationship strength is compared with a threshold, if the relationship strength is higher than the threshold the two people are friends, or they are strangers. In the protocols we mentioned in the thesis, researchers always assume that friends are trustful. That is reasonable because you might feel more secure when your friends forward your message instead of a stranger.

2.5 Location-privacy protocols

The basic idea of most location-privacy protocols is hiding the original requester behind a set of users so that attackers can hardly infer the identity of the original requester. In other words, when a user sends a query, his identity cannot be inferred based on the information in the query easily. For example, the original requester can use a pseudonym instead of his own name or use an obfuscated location to send his queries, so that attackers cannot tell the different between the queries and a set of other users' queries.

The location-privacy protocols are considered as centralized and distributed ones based on whether they require any infrastructure.

2.5.1 Centralized Protocols

Centralized Protocols always need some infrastructure to serve for the obfuscation process. The infrastructure could be any equipment which act as anonymization servers. When users want to send a query to the LBS server, he sends the query to the anonymization server. The anonymization server changes the identity and the location in the query, which is called anonymization or obfuscation. Then the anonymization server forwards the modified query to the LBS server, so that the attacker who attacks the LBS server can hardly learn the identity of the original requester. The LBS server can send the reply message to the anonymization server which is responsible and able to forward the reply message back to the original requester. In this way, the original requester gets the information he needs, while avoiding from being located by attackers.

The advantage of the centralized protocols is that the anonymization server has more resources and information than users in the network, which enable the server to achieve better anonymization performance and provide sufficient protection for users. For example, if the anonymization server knows locations of all users in the network, it can modify the location and identity in the queries to the most suitable, with which the LBS server can provide acceptable results while the original requester is not exposed.

There are three major defects of this kind of protocol. 1. It is hard to deploy infrastructure in some area; 2. Since the anonymization server knows users' location, it is also a risk for users.

2.5.1.1 Single Server

Authors in [9] uses a central anonymity server through which the mobile users can send queries to external services. To make the communication between users and the central anonymity server trustful, users set up an encrypted connection with the anonymity server at the beginning. Users sends their encrypted queries to the anonymity server, so that the anonymity server is the only one who can learn the information in the query. The anonymity server decrypts the queries and uses a cloaking algorithm to perturb the position information in the queries. Then the anonymity server sends the modified queries to external services (e.g. LBS). This is a typical centralized protocol, which can reduce the re-identification risk for users. Its defect is that a continuous connection to the server is necessary for each user, which is hard to achieve in a sparse DTN. We cannot image that a MANET user can have a stable connection with a central anonymity server, neither.

In [10], researchers employ a matchmaker which is used to match users and advertisements, then users can achieve anonymization of their identities and locations from the matchmaker. The system architecture in [10] is similar to the former [**?**], while authors in [10] just focus on the advertisement service instead of the ?external services? in [10]. The role of matchmaker is matching users and advertisements. Although the functions of the matchmaker in [10] and the central anonymity server in [9] are different, the intermediate trusted three-party servers (i.e., the matchmaker and the central anonymity server) separate users and the application server (e.g., an advertisement server, LBS server and so on).

The architecture in [10] does not require an encryption process so that attackers can learn the information in the queries. Although a plenty of queries arrive the matchmaker in a short time from different users also help the matchmaker mix the queries so that attackers can hardly trace the query, the burden of the users who are near the matchmaker is heavy.

In the work in [14], researchers use ?a trusted, third-party location anonymization engine (LAE) that acts as a middle layer between mobile users and LBS providers? [14], so that exact locations and requests from clients are replaced by a location anonymization engine before they arrive at LBS providers. Therefore, we learn that the structure of this kind of algorithms is almost like Figure 2.5. A trusted server is placed between users and the application server, which hides users' information and offers sufficient information for the application servers to provide acceptable services.

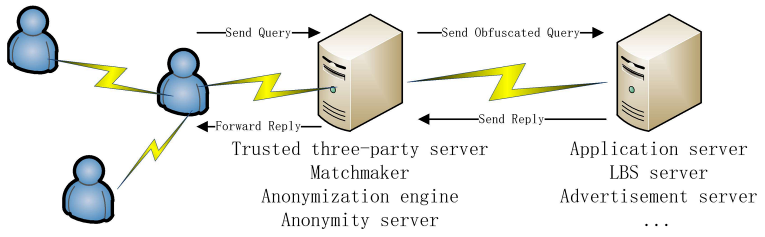


Figure 2.4: Single Anonymization Server

2.5.1.2 Multiple Servers

The above protocols always use a single anonymization server, while there are also other protocols which need more infrastructures.

Authors in [8] propose a protocol, called Social-based PRivacy-preserving packet forwardING (SPRING), for vehicular delay tolerant network. In their work, they employ Roadside Units (RSUs), which are a type of equipment deployed along the roadside, to assist the packet forwarding and achieve conditional privacy preservation. These RSUs are located at high social intersections, so that vehicles which pass by the RSUs send their messages to the RSUs. The RSUs have sufficient resource so that they can hold the messages for a long time, which decrease the probability that the messages are dropped. The messages are forwarded to proper next-hop vehicles when the vehicles pass by the RSUs. Since messages are hold by the RSUs for a period, attackers can hardly trace messages. Besides, a large number of vehicles send many packets to these RSUs, which enables the RSUs to serve as mix servers. The advantage of SPRING is that it improves the delivery success ratio and privacy-protection performance. However, deploying the RSUs is not always feasible.

Another example is the work in [4]. Researchers use sensor nodes which are scattered throughout the network to provide anonymized locations for users, as shown in Figure 2.6. The whole system area is partitioned into a set of aggregate locations by the sensor nodes, where there are at least k persons. To provide better location-based services, they minimize the areas of aggregate locations. When a user sends a query to the LBS, he uses the location of the sensor nodes which he belongs to, so that attackers cannot tell the difference between the requester and the other $k-1$ users in that aggregate location. This is a typical k anonymity algorithm, while its disadvantage is that it is difficult to deploy the sensor nodes in real-world. Besides, the mix servers and sensor nodes might be more prominent targets than LBS providers.

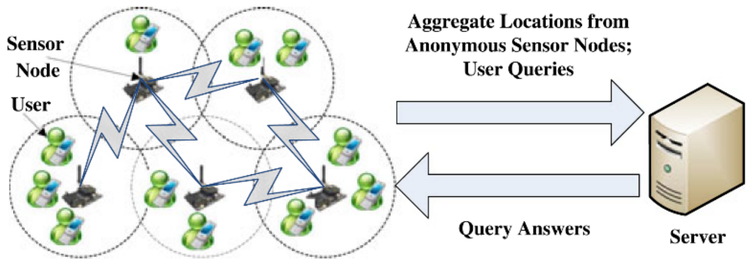

Sensor Node
User
Aggregate Locations from Anonymous Sensor Nodes; User Queries
Query Answers
Server

Figure 2.5: REAL [8]

2.5.2 Distributed Protocols

Although centralized protocols also have good performance in their delivery success ratio and privacy-protection in their own field, the MANET is a network without infrastructure. In general, it is not allowed that deploy any infrastructure for MANET. Distributed protocols are organized by users independently so that they are more proper for applications in MANET. A problem for a distributed protocol it that whether a user can trust another one. Authors in [16], [17] and [18] introduce the social tie to determine whether a user is trustful.

In the distributed social based location privacy protocol (SLPD) [16], authors import 2 concepts: the obfuscation phase and the free phase. A query from an original requester is initialed as the obfuscation phase, then it must be transmitted between friends for k times. For example, the original requester sends the query to one of his friend in one hop, then the friend forwards the query to another friend. We call these friends the agents. That process repeats for exact k times. When the k^{th} friend get the query, he switches the query to the free phase and replaces the sender identity with his own identity, then sends the query to the destination (e.g., LBS) with any DTN protocol. The LBS sends the reply to the friends and they are responsible to forward the message to the original requester.

In this way, attackers can only learn the identity of the last friend instead of the original requester. Since each user who knows the identity of the original requester is the original requester's friend who are assumed trustful, there are no attackers among these agents. Therefore, attackers can hardly learn the identity of the original requester.

The disadvantage of the protocol is that it is hard to encounter a friend in the network, which decreases the success ratio of delivering queries. The how to improve the success ratio is a key of these protocols.

In Hybrid and Social-aware Location-Privacy in Opportunistic mobile social networks (HSLPO) [17], authors try to improve the delivery performance by using a stochastic model which uses a Markov model for location predication. The major process is similar with SLPD, but an agent can forward the query a user who is not a friend of the original requester if the user has more chance to deliver the query and a trust value larger than a threshold. In other words, an agent continuously searching his surrounding to find the original requester's friends. If the agent cannot find the original requester's friend but his own friend, he checks whether his friend has more chance to deliver the query than him using the Markov model. If his friend is more proper for forwarding, he sends the query to his friend. The performance of HSLPO depends on the Markov model, that is, whether it can predict users' movement accurately. In our real-world, people's movement model is more complicated than that in experiment.

Another protocol, called Location Privacy-Aware Forwarding (LPAF) [18], also attempt to improve the performance of the SLPD. The LPAF and the SLPD are similar, while the LPAF just add more friends to the protocol. When an agent cannot find a close friend (i.e., their trust value is high), he will try to find other general friends (i.e., their trust values is lower than the close friends). That might be a safety tradeoff, because some ineligible users in SLPD are chosen as friends based on the additional standards imported by LPAF.

In fact, both the HSLPO and LPAF cannot solve the problem in finding friends. Although we use the friends of friends or set the threshold for friends lower, there are only a few users can be chosen as agents. In LPAF, the identity of the requester is even exposed in front of someone who are not very trustful.

Chapter 3

Multi-Hop Location-Privacy Protection

3.1 System Model

Our network architecture consists of two main entities: Users and LBS Providers (LBSPs). Due to the introduction of the social network, the users' social information can be used in obfuscation forwarding process. Based on available information in the social network, the relationship between two users can be considered as friends or strangers. The user who makes a query to an LBSP will be called the original requester while the others are called intermediate users. LBSPs are located in fixed locations and their coordinates are known by all users when users join the network. Attackers are assumed to be able to access LBSPs, and attempt to locate original requesters. We assume that the LBSPs are semi-trusted and the strangers are un-trusted. We also assume that both entities have sufficient resources, like computational capability, storage and battery power.

Since two friends could be a pair of multi-hop neighbors, users can leverage Optimized Link State Routing Protocol [29] to seek friends continuously after entering the network, so that they can recognize each other and make contact in time. When a user carries an obfuscation phase query, he might send the query to a multi-hop friend through several strangers. In this case, a secure communication is necessary for between them, so that they must send the query encrypted to prevent strangers from learning anything about the query. Each user obtains a pair of asymmetric keys (public and secret key) before he joins the network from a certificate authority using well-regarded techniques, like in [28]. Whenever a user detects a new friend, he sends a request to the friend asking for his public key. In this way, a user can get his friends' public key when they encounter each other. Even though several strangers can be active in the obfuscation phase, the queries can still be securely sent to the user's friend.

The relationship strength is often "a hidden effect of nodal profile similarities" [30]. Let $SV_{i,j}$ denote a value of relationship strength which user i determines whether user j is an acceptable friend based on the relationship strength. For every pair of users (i and j), we assume that there is an $SV_{i,j}$. If $SV_{i,j}$ is bigger than a specific friend threshold T_{min}, set by the original requester, user j is considered as a friend of user i; otherwise, it will be treated as a stranger. The notations used in this paper and their meanings are shown in Table 3.1.

3.2 Details of MHLPP

MHLPP aims to protect the original requester's (N_0's) location-privacy using an obfuscation path. In other words, a query q which needs to be obfuscated must go through a series of friends after it leaves N_0. The whole process includes two parts: the obfuscation phase and the free phase. In the former phase, q is only transmitted among friends, until it is sent to an area called "$obfuscationarea$". At the end of that phase the last friend N_f replaces all N_0's information by its own and forwards q with an arbitrary DTN forwarding protocol, like the one suggested in [31]. In this case, what attackers can learn from the

Table 3.1: MHLPP Symbols

Parameter	Meanings
N_0	the original requester
N_i	if $i > 0$, it denotes the friend chosen by N_{i-1}. If $i = 0$, it is N_0.
N_f	the last friend who handles the obfuscation query
N_d	the destination or the LBSP
K_i	the public key of N_i
S_i	the secret key of N_i
q	a query of N_0
r_q	the requirement for the query q
msg	a message contains q and r_q
$Emsg_i$	the encrypted msg using K_i
S_{id}	the original requester's identity
D_{id}	the destination's identity
L_s	the location of N_0 when it sends the query to N_1
R_p	the inner radius of the obfuscation area
R_s	the external radius of the obfuscation area
T_{min}	the social value bound for friends
C_{max}	the extra path limit in each obfuscation forward

database in LBSP is N_f's information, so they can hardly infer the original requester's identity and location based on that information. The free phase starts when the query q is forwarded by N_f and ends when it reaches the LBSP.

Because N_f is the only identity the LBSP knows, the LBSP has no choice other than replying to the last friend N_f when it receives the obfuscated query. The reply can be delivered with an arbitrary DTN routing protocol as the free phase query does. The friend N_f should remember who is the real destination (N_0) of this reply, then he transmits it to N_0. In this way, N_0 is able to send a query q to an LBSP while not exposing his own information.

The obfuscation phase of a query q starts when the query leaves the original requester N_0. When a user is holding an obfuscation phase query, he starts sensing connected friends continuously, which enables it to communicate with one-hop or multi-hop neighbor friends. Even though users in the mobile network use OLSR protocol [29] to detect others automatically, they do not communicate with their friend unless they have a requirement to send an obfuscation query. Also, they do not ask their friends for public keys. Therefore, carrying an obfuscation phase query requires a user to execute MHLPP algorithm.

When N_0 finds the first available friend N_1, he asks N_1 for a public key K_1, which will enable him to encrypt his query using K_1. That prevent others, e.g. strangers, from learning information in the message msg that that contains both q and r_q. r_q is N_0's requirement for q, which is always sent with the query q and remains constant until the end of the obfuscation phase (we discuss this in part C). Friends who get the query can infer q's obfuscation area based on parameters R_p, R_s and L_s in r_q, which is a ring with inner radius R_p, external radius R_s and center L_s. Before N_0 sends the query q to his friend, he initializes parameter L_s to his current location and encrypts msg using K_1 to get $Emsg_1$ which is what N_0 sends to N_1.

The destination of $Emsg_1$ is N_1, which is a plaintext in $Emsg_1$, so that other intermediate users (strangers) can help N_0 forward $Emsg_1$ to N_1. In this step, strangers transmit $Emsg_1$ using the OLSR protocol if N_1 is a multi-hop neighbor of N_0. Strangers learn nothing other than the identity of N_1, because both q and r_q are encrypted. They cannot help attackers locate N_0 because they do not know S_{id} (the identity of N_0) and D_{id} (the LBSP) included in r_q.

When N_1 receives $Emsg_1$, he decrypts it with its secret key S_1 to get q and r_q. If q is already in the obfuscation area defined in r_q (i.e., it is already in the ring), the query q finishes its the obfuscation phase. Then N_1 replaces all information of N_0 with his own. For example, the S_{id} is replaced with N_1. If a location is necessary for the LBS, N_1 uses his own current location and records this change in his memory before initiating the free phase. A free phase query can then be forwarded to the destination (i.e. LBSP).

If q is still in the obfuscation phase (not in the ring), N_1 performs similar actions just as N_0 expect modifying r_q. Another difference is that instead of finding friend randomly,

N_1 would seek for a friend who is nearer in the obfuscation area, and so will the following friend.

The detailed algorithm is explained in Algorithm 3-1 where N_x is a neighbor of N_i who is carrying the query q. Function "*DealWithQuery*" is responsible for dealing with a query. For N_0, he generates the query q and its requirement r_q. If N_i receives $Emsg_i$, he decrypts it with his own secret key S_i. If q finished its obfuscation phase at N_i, q will be required to be forwarded in free phase immediately. Otherwise, q needs to be processed in the obfuscation process. Both q and r_q are stored in N_i until they are sent to the next friend. N_i starts detecting friends continuously if and only if N_i carries one or more obfuscation phase queries.

When N_i detects a new neighbor N_x (one-hop or multi-hop neighbor), he follows steps in "*WhenEncounterUser*". For an expired query, N_i simply drops it. If the query q is already inside its obfuscation area, N_i switches it to the free phase. If q stays in the obfuscation phase and N_x is an available friend, N_i encrypts both q and r_q using N_x's public key K_x to get an encrypted message $Emsg_x$. Then N_i forwards $Emsg_x$ to N_x and stops sensing friends after $Emsg_x$ departs from it.

When we mention that N_i switches a query q to the free phase, N_i actually replaces N_0's information with its own one in q to get q^* and records this replacement in its storage, then N_i uses the Spray and Wait protocol [31] to forward q^* in plaintext. That allows N_i to hide N_0's identity and forward a reply from LBSP to N_0.

3.3 Requirement parameters

In the obfuscation phase, r_q is always in msg so that friends who get msg can make decisions (e.g. selections of friends) based on it.

All parameters (i.e., S_{id}, D_{id}, R_p, R_s, L_s ,T_{min} and C_{max}) in r_q are given by N_0 before

Algorithm 1 The Obfuscation Phase in MHLPP

```
 1: procedure DealWithQuery
 2:     if the current user is $N_0$ then
 3:         $generate\,(q, r_q)$ by himself
 4:     else
 5:         if the current user is $N_i$, $i > 0$ then
 6:             $(q, r_q) \leftarrow Decrypt_{S_i}\,(Emsg_i)$
 7:     if $q$ can switch to the free phase based on $r_q$ then
 8:         $SwitchFree\,(q)$
 9:     else
10:         $msg \leftarrow (q, r_q)$
11:         Store $msg$
12:         Start sensing friends
13: procedure WhenEncounterUser($N_x$)
14:     $q \leftarrow$ get query from $msg$, $r_q \leftarrow$ get requirement from $msg$
15:     if $q$ timeout then
16:         remove $msg$ return
17:     if $q$ is eligible to switch into the free phase then
18:         $SwitchFree\,(q)$ return
19:     if $SV_{i,x}$ is bigger than $T_{min}$ in $r_q$ then
20:         if it is the first hop of $q$ then
21:             assign the current location to $L_s$ of $r_q$
22:         $Emsg_x \leftarrow Encrypt_{S_x}\,(q, r_q)$
23:         forward $Emsg_x$ to $N_x$
24:         remove $msg$ from memory
25:         stop sensing friends
26: procedure SwitchFree($q$)
27:     $q^* \leftarrow$ replace $q$'s requester-information $(N_0)$ by $N_i$
28:     record $(q, q^*)$
29:     forward $q^*$ with DTN protocols
```

$Emsg_1$ leaves N_0. Parameters S_{id} and D_{id} record the identities of N_0 and the destination (LBSP) N_d, based on which last friend N_f is able to send the query freely to D_{id} (N_d) and forward the reply to S_{id} (N_0).

The obfuscation area is a ring with an inner radius R_p and an external radius R_s. As shown in Figure 3.1-a, the obfuscation area is actually the grey area "a". Obfuscation area must guarantee both the original requester's location-privacy and location awareness. In other words, the value of R_p should be big enough, so that there are sufficient users in the inner circle (with a radius R_p). At the same time, R_s should be small enough so that the LBSP can provide a service, acceptable to N_0.

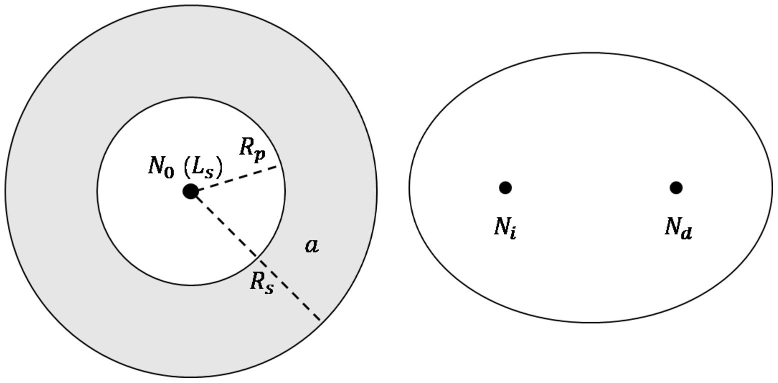


Figure 3.1: The selection of the next friend

A user N_x can be chosen by N_i as a friend for q if and only if $SV_{i,x}$ is bigger than the threshold T_{min}. The original requester N_0 can set various values for his queries based on their importance. If T_{min} is large, there would be fewer friends for any users in the network which reduces the query success rate to a certain extent, as a result. We assume that the original requester can balance the level of privacy and the success ratio.

Most DTN routing protocols aim to deliver queries through the shortest path, while MHLPP pays more attention to security in its obfuscation phase. Consequently, the obfuscation process in MHLPP results in a longer path from the original requester N_0 to the destination N_d. To limit the length of the path, we introduce parameter C_{max} which is the maximum extra path (e.g., the difference of the length and the distance between the N_i

and LBSP) we can tolerate. For any friend N_i who gets a C_{max} from r_q, if he selects N_x as the next friend, the extra path should not be longer than C_{max}. Let's denote the optimal path from user m to n by $Dis\,(m, n)$, and the extra path from N_i to N_d through N_x by $C_{i,x,d}$. Then $C_{i,x,d}$ can be defined as follow.

$$C_{i,x,d} = Dis(i,x) + Dis(x,d) - Dis(i,d) \tag{3.1}$$

$C_{i,x,d}$ must be a value smaller than C_{max}. If $Dis\,(m,n)$ is the straight-line distance between point m and n, then next friend N_x should be in an ellipse E_C with focus points N_i and N_d. Let's denote the coordinate of N_i by $\left(-\frac{d}{2}, 0\right)$ and the coordinate of N_d by $\left(\frac{d}{2}, 0\right)$. Then, the equation of the ellipse E_C is

$$\frac{x^2}{(d + C_{\max})^2} + \frac{y^2}{2d \cdot C_{\max} + C_{\max}^2} = \frac{1}{4} \tag{3.2}$$

As shown in Figure 3.1-a, L_s is the center of the ring while R_p and R_s are the inner and external radii, respectively. The query q switches to the free phase when it enters the ring area.

As shown in Figure 3.1-b, N_i who is carrying obfuscation queries should choose his next friend N_x in the ellipse, which avoids the query q going through an unacceptably long path.

In conclusion, a query q starts at the center and moves inside the ring, until it reaches the obfuscation area. The point N_i in Figure 3.1-b should be inside the ring, and the point N_d might be anywhere. As a result, a user N_i who is carrying an obfuscation query detects a friend continuously who has a larger distance from L_s and inside an ellipse. If there is a friend like that, N_i sends the query to that friend N_x.

3.4 Privacy Analysis

We assume that attackers can achieve all information in LBSPs know. Obviously, they can know the identity of the last friend N_f who replaces N_0's information with his own. It is possible for the attackers to locate N_f with little cost. For example, if N_0stops moving after sending the query q, it is reasonable for N_f to believe that N_0 is in a ring centered at the location of itself with radii R_p and R_s. In other words, the distance between N_f and N_0 should be in a range between R_p and R_s. If attackers find all users who satisfy this condition, the original requester might be among these users with high probability. Then, a success ratio $P_{r_p r_s}$ to locate N_0 can be measured by a conditional probability

$$P_{r_p r_s} = p_{r_p r_s} \cdot \frac{1}{m_{r_p r_s}} \tag{3.3}$$

where $P_{r_p r_s}$ is the probability that N_0 is in the ring (i.e., the distance between N_0 and N_f is larger than R_p and smaller than R_s). Here, $m_{r_p r_s}$ is the number of users who are in the ring. Attackers locate N_0 successfully if and only if N_0 is in the ring, at the same time, attackers pick the correct one from all $m_{r_p r_s}$ users at that area.

In the worst case, attackers know exact values R_s and R_p. Then, the Eqn. (3.3) becomes

$$P_{R_p R_s} = p_{R_p R_s} \cdot \frac{1}{m_{R_p R_s}} \tag{3.4}$$

where $P_{r_p r_s}$ is the probability that N_0 is on the ring (i.e., the distance between N_0 and N_f is larger than R_p and smaller than R_s). $m_{r_p r_s}$ is the number of users who are in the ring. Since parameters (e.g., R_p and R_s) in r_q are kept secret among trusted friends in our system model, attackers can hardly get the actual values of those parameters.

3.5 Complexity discussion

In order to guarantee secure communications among friends, encryption is introduced in our protocol. In the obfuscation phase, the query is transmitted along friends, i.e. N_0, N_1, N_2, ..., N_f. When the query is sent from N_i to N_{i+1}, a pair of encryption and decryption is needed, so the number of such pairs T_{en} should be equal to f.

T_{en} grows with both T_{min} (threshold used to decide friend relationship) and inner radius R_p. Essentially, it is the number of friends participating in transmitting a query q in its obfuscation phase that influences T_{en}. Given a smaller T_{min}, a user carrying the obfuscation phase query has more chances to encounter more friends in a certain area. A larger R_p also leads to a bigger area inside the ring, so that there are more friends in this area. We evaluate the number of encryptions and decryptions in our simulation. Figure 3.2 shows the average number of encryptions (T_{en}) with different friend thresholds T_{min} and inner radius R_p. We observe that T_{en} increases steadily as we increase T_{min} and R_p.

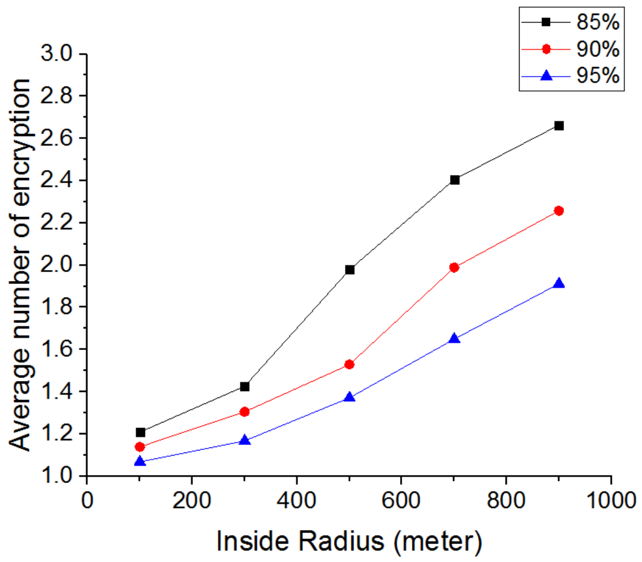


Figure 3.2: The number of encryption with various inner radius

It is evident that encryption and decryption process in MHLPP results in an extra cost in both energy and computational resources. However, the number of encryptions and

Table 3.2: MHLPP Experiment Parameters

Parameter	Value
Simulation Time	10 minutes
Map Size (W x H)	4500 m x 3400 m
Total number of users	126
Pedestrians/ Cars	84/42
Communication Area Radius	10m – 90 m
Pedestrian Speed	1.8-5.4 Km/h
Car Speed	10-50 Km/h

decryptions is quite low (below 3) which reasonable based on the experiment.

3.6 Performance Analysis

We use the map of Helsinki in our simulator to evaluate MHLPP. It is also compared against the known protocol, Hybrid and Social-aware Location-Privacy in Opportunistic mobile social networks (HSLPO). The simulation parameters are shown in Table 3.2. All pedestrians and cars are users in MHLPP. These users are moving on the map along streets continuously. There is an LBSP fixed at a random location on the map. For each user, we give him random social values between 0% and 100%, each corresponding to all other users. Each value has the same probability, so we can compute the expected number of friends of a user. For example, if we are given a privacy threshold (T_{min}) of 85%, then there might be 15% (100%-85%) users who are friends of a certain user.

As shown in Table 3.2, there are 126 users in the map. For each of them, say user i, we give him 126 random SV values, which denotes the relationship strength between him and other users, so that there are 126^2 SVs in our simulation. The SVs are between 0 and 100. As a result, if the T_{min} is equal to 85, the average number of friends of each user should be 18.9 (=$126 \times (100 - 85)/100$).

Users are placed at random locations at the beginning of each experiment. We choose another random point for each user, so that he can move back and forth along streets between that point and the point his starts at. The user speed depends on its type (pedestrians or cars) and set randomly. All queries have a 10-minute timeout. The queries which are expired before they reach the LBSP (the destination) are considered to be failed in our success ratio statistics.

Figs. 3-5 compare performances between HSLPO and MHLPP for different values of k, communication radius and privacy threshold (T_{min} in MHLPP). The k is the privacy-level requirement in HSLPO. Both HSLPO and MHLPP have different criteria in which a query can switch to the free phase. To make them comparable, we create a new parameter, called *obfuscation distance*. If a query leaves N_0 at location L_a and switches to the free phase at N_f whose location is L_b, then the obfuscation distance is the straight-line distance between L_a and L_b. We test the obfuscation distances of HSLPO with different parameters, and then we set the inner radius of MHLPP to those values. The query success ratio is the ratio of delivered queries to the total number of queries. The number of hops (h) is the number of intermediate users between N_0 and the destination (LBSP). We count the number of users surrounding the last friend in a specific range, which is k times the communication radius. We calculate the entropy using the reciprocal of the number of surrounding users.

3.6.1 Query success ratio

The query success ratio is the percentage of delivered queries among a number of attempts. Based on the timeout value in Table II, a query is delivered successfully, if it arrives at the LBSP (the destination) before the timeout; otherwise it fails. We use the query success ratio to evaluate the delivery performance of MHLPP.

As shown in Figure 3.3-a, the success ratio in MHLPP is always higher than that in HSLPO. As the value of k increases, HSLPO success ratio drops sharply while MHLPP remains stable. This is because the larger k is, the harder it is for HSLPO to find enough friends in a limited time. The lack of friends has less impact on MHLPP. We observe that the success ratio of MHLPP rises when $k = 7$. That is because it depends on the inner radius which is equal to the obfuscation distance of HSLPO. The obfuscation distance decreases when $k = 7$, because most of the queries which complete their obfuscation phase have a short obfuscation distance. In Figure 3.3-b, both HSLPO and MSLPP values increase and have the same trend when given a larger communication radius. The reason is that the communication radius effects the free phase more than the obfuscation phase for both. As shown in Figure 3.3-c, higher privacy threshold leads to lower success ratio in two algorithms. Its impact on HSLPO is more intense than that on MHLPP, which is the most important characteristic of MHLPP. MHLPP has a better performance than HSLPO especially when there are fewer friends in the network. MHLPP can transmit messages with the help from strangers in its obfuscation phase while HSLPO cannot.

3.6.2 Number of Hops

We count the number of hops it takes for queries to be delivered successfully and calculate the average. Every user who takes part in the delivery process is considered in the hop count. We introduce this criterion to measure the routing path length of the algorithm. MHLPP is more sensitive to the probability that a user encounters a friend than HSLPO is. The reason is that MHLPP aims to reach a certain distance rather than taking certain number of hops. In other words, MHLPP continues sending queries to other friends until queries enter their obfuscation area. In this process, MHLPP takes every chance to forward queries. If it is hard for MHLPP to find friends, it can also take the queries to obfuscation areas with fewer friends. Therefore, the probability that users encounter friends has less impact on the performance of MHLPP. The result of this experiment is shown in Figure 3.4.

In Figure 3.4-a, the number of hops in HSLPO is affected by parameter k obviously. Especially, the first k hops forwarders must be friends, which makes it hard for HSLPO to have queries to be forwarded successfully. Both protocols have similar number of hops in their free phases, but HSLPO has exactly k hops in its obfuscation phase, while MHLPP can have fewer than k hops. In figure 4b, the number of hops in MHLPP grows with the communication radius obviously, while it does not change a lot in HSLPO, because only successful delivery queries are counted in the statistics. Given a large communication radius, MHLPP has a much higher success ratio for delivered queries as it can connect more friends. In Figure 3.4-c, the value of MHLPP drops for higher privacy thresholds. The reason is the same as Figure 3.4-b. For HSLPO, no matter how hard it is to find a friend, it attempts to find exactly k friends. However, if it is too hard to find a friend, MHLPP's friend can carry the query while moving and complete the obfuscation process. For higher privacy thresholds (i.e. resulting in fewer friends), MHLPP chooses to carry queries other than finding friends. That results in a drop in the number of hops.

3.6.3 Security

Since the principles with that two protocols protect original requesters are different, we evaluate the probability that attackers locate the original requester if the distance between him and the last friend is smaller than a some value r, which is equal to k times the communication radius. Since the last friend reveals himself to the LBSP, attackers might locate him accurately. We assume that attackers know the privacy parameter k. In the worst case, the distance between the original requester and the last friend is smaller than r when attackers start to locate the original requester. That gives attackers a chance to locate the original requester. We count all users who are inside the radius r of the last friend, and the original requester is one of them. For example, if there are m users in the area, the probability should be $\frac{1}{m}$. Figure 5 compares the entropy E of both HSLPO and

MHLPP. We use the following formula for computing entropy:

$$E = -\frac{1}{m} log_2 \left(\frac{1}{m} \right)$$

From Figure 3.5-a, we observe that MHLPP has very small (about 0.04) increase in entropy compared to HSLPO. When the original requester is in the circle centered at the last friend, HSLPO is a little more secure than MHLPP but not by very much. That is because HSLPO always switches to the free phase when the last friend encounters the previous friend, so the previous friend must in the circle. MHLPP does not have this condition. Both graphs in Figure 3.5-a and Figure 3.5-b have the same trend. The curve of MHLPP is also a little higher than that of HSLPO, while two curves almost meet when the communication radius is small or large. Given a small radius, the entropy of both protocols are small. When the radius is large, we can ignore the effect of the previous friend mentioned about for Figure 3.5-a as there are so many users in the circle. From Figure 3.5-c, since the circle neither expands or shrinks, we observe that the two curves exhibit similar behavior. The values of HSLPO are always lower than the correlated values of MHLPP. However, as we observed in the experiment, the last friend is always hundreds of meters away from the requester.

References

[1] L. J. Chen, C. H. Yu, T. Sun, Y. C. Chen, and H. H. Chu. A hybrid routing approach for opportunistic networks. In *the 2006 SIGCOMM workshop on Challenged networks (CHANTS)*, pages 213–220, September 2006.

[2] M. Duckham. Moving forward: Location privacy and location awareness. In *the 3rd ACM International Workshop on Security and Privacy in GIS and LBS*. ACM, November 2010.

[3] K. Fall. A delay-tolerant network architecture for challenged internets. In *the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 27–34, 2003.

[4] C. Chow J. Zhang. Real: A reciprocal protocol for location privacy in wireless sensor networks. *IEEE Transactions on Dependable and Secure Computing*, 12(4):458–471, October 2014.

[5] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. The one simulator for dtn protocol evaluation. In *the 2nd International Conference on Simulation Tools and Techniques*. ICST, March 2009.

[6] John Krumm. A survey of computational location privacy. *Personal and Ubiquitous Computing*, 13(6):391–399, August 2009.

[7] Y. Liao, K. Tan, Z. Zhang, and L. Gao. Estimation based erasure coding routing in

delay tolerant networks. In *the 2006 international conference on Wireless communications and mobile computing (IWCMC)*, pages 557–562, July 2006.

[8] R. Lu, X. Lin, and X. Shen. Spring: A social-based privacy-preserving packet forwarding protocol for vehicular delay tolerant networks. In *2010 Proceedings IEEE INFOCOM*. IEEE, March 2010.

[9] D. Grunwald M. Gruteser. Anonymous usage of location-based services through spatial and temporal cloaking. In *the 1st International Conference on Mobile Systems, Applications and Services*, pages 31–42, May 2003.

[10] M. Mano and Y. Ishikawa. Anonymizing user location and profile information for privacy-aware mobile services. In *the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, pages 68–75. ACM, November 2010.

[11] Pietiläinen and A.K. *Opportunistic Mobile Social Networks at Work*. PhD thesis, Université Pierre et Marie Curie, Paris, 2010.

[12] E. Reddy, S. Kumar, N. Rollings, and R. Chandra. Mobile application for dengue fever monitoring and tracking via gps: Case study for fiji. Technical Report TR-04-2015, Mar 2015.

[13] J. Schiller and A. Voisard. *Handbook of wireless networks and mobile computing*. Morgan Kaufmann, 2004.

[14] L. Liu T. Wang. Privacy-aware mobile services over road networks. *the VLDB Endowment*, 2(1):1042–1053, August 2009.

[15] R. Xiang, J. Neville, and M. Rogati. Modeling relationship strength in online social networks. In *the 19th International Conference on World Wide Web*, pages 644–654. ACM, April 2010.

[16] S. Zakhary and M. Radenkovic. Utilizing social links for location privacy in opportunistic delay-tolerant networks. In *International Conference on Communications (ICC)*. IEEE, June 2012.

[17] S. Zakhary, M. Radenkovic, and A. Benslimane. The quest for location-privacy in opportunistic mobile social networks. In *Wireless Communications and Mobile Computing Conference (IWCMC)*. IEEE, July 2013.

[18] S. Zakhary, M. Radenkovic, and A. Benslimane. Efficient location privacy-aware forwarding in opportunistic mobile networks. *IEEE Transactions on Vehicular Technology*, 63(2):893–906, February 2014.

[19] M. M. Zanjireh and H. Larijani. A survey on centralised and distributed clustering routing algorithms for wsns. In *Vehicular Technology Conference (VTC Spring)*. IEEE, May 2005.