1-1-2012

Location Privacy In Emerging Network-Based Applications

Yong Xi

Wayne State University,

LOCATION PRIVACY IN EMERGING NETWORK-BASED APPLICATIONS

by

YONG XI

DISSERTATION

Submitted to the Graduate School

of Wayne State University,

Detroit, Michigan

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

2012

MAJOR: COMPUTER SCIENCE

Approved by:

_____

Advisor Date

_____

Co-Advisor Date

_____

_____

_____

ACKNOWLEDGMENTS

Working on the Ph.D. has been a good experience. The topics themselves and the learning/exploring experience are inseparable parts of this process that would prove invaluable in the days to come. I would like to express my deepest appreciation to my committee members, friends, and family. Without their guidance, help, and support, I would not have been able to finish my dissertation.

First of all, I am deeply grateful to my advisors Professor Loren Schwiebert and Professor Weisong Shi. Professor Schwiebert has been a patient mentor that tirelessly guides me through different stages of my study and across a domain of problems that I had worked on. His wisdom and experience has always lended invaluable insights. I thank him for the encouragement and financial support that has sustained my research. Professor Shi's enthusiasm for research has greatly affected me. I thank his guidance in navigating the problem domain and comments on the joint work of privacy with Professor Schwiebert.

I also thank the NSF and department of computer science for providing financial support for my research and studies at Wayne State University. I would also like to thank many people at department of computer science for their general support.

Special thanks to Dr. Tao Zhang for the joint work on privacy in vehicular networks. Also, I would like to thank Kewei Sha, Zhengqiang Liang, Manish Kochal, Fernando Martincic, and other colleagues for the discussions over years on topics from research problems, methodologies, and Ph.D. studies.

I would also like to thank my parents, two elder sisters, and elder brother for their encouragement.

Finally, I thank my wife, Jin Luo, for her encouragement, understanding, and patentience.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

The respecting of individual rights is a tradition in American society. The right to privacy is a derivation of this individual liberty. As the society has transformed into an information society, information privacy is defined as one's ability to control the flow of private information. Although the definition over what information is private varies under different scenarios, it is generally agreed that privacy is more or less a concern.

As information is a very general word, the actual meaning of privacy varies widely depending on the specific context. Similar as security, privacy is a system property and extends into social sciences. In this chapter, we first try to provide a big picture on general privacy research in computer science. Then we proceed to discuss location privacy and our research topic.

1.1 State of art in privacy research

In technical terms, one's ability to control the flow of private information must rely on the underlying mechanisms provided by the computer and network system in an information society where information is processed and moved in the computer and network system. Thus, similar to security, privacy is an aspect of such a system.

Within the field of computer science, privacy is investigated under roughly three contexts: information system, operating system, and networking.

1.1.1 Privacy research in information systems

One of the major systems today is the Internet. People conduct various activities online with different types of websites, such as e-commerce websites, web-based email services, and social network websites, etc. In order to deliver personalized services to individuals, users are generally required to disclose certain personal information to the providers. User privacy can be addressed in two ways. The first is from the client side. Specifically, what information is necessary for certain personalization? What are the implications of disclosing such information? Kobsa [49] discussed several factors affecting user privacy. The main conclusion is that privacy perceived by users is diversified and can not be guaranteed by a single approach. Thus numerous privacy-enhancing techniques need to be built into a system gradually over time.

The second is from the organization side. Specifically, what is the privacy policy governing the contract between the user and the organization? How is the user information being used inside an organization? The use of a privacy policy has several problems. First, organizations tend to use natural languages to describe their privacy policies. However, the language used often has multiple meanings as described by Pollach [69], "companies obscure privacy infringements by downplaying their frequency, mitigating or enhancing questionable practices, and omitting references to themselves when they talk about unethical data handling practices." Second, there is no existing enforcement and auditing of privacy policies within organizations. Thus, Antón *et al.* propose a framework for comprehensive online privacy management [7]. In their proposal, the natural language based privacy policies are mapped into security policies governing authentication, access control, and information flow. The security policies in turn are realized through lower layer mechanisms. For example, the access rights to a relational database are not only defined on the table or column level but also defined on the row level or even cell level.

A fundamental research aspect on privacy in information systems is associated with databases. Besides the fine-grained access control mentioned above, a large portion of research have been dedicated to enhancing privacy in statistical databases [26, 19] and privacy-preserving data mining [5, 88].

1.1.2 Privacy research in operating systems

In its traditional role, the operating system sits between applications and hardware, and provides isolation between applications. This leads to a natural extension to a privacy-enhanced operating system. However, in traditional operating system designs, privacy is not built-in by design. For example, Singh *et al.* showed that under the original access control *(owner, group, others)* models in *NIX systems, data shared between two users are easily breached by simple attacks [79]. This is mainly due to the tree structured permission management and lack of support for effective auditing.

In order to fix the lack of explicit privacy support in operating systems, it is necessary to modify the operating system itself. Singh [79] introduced a view-based access control layer on top of Linux file systems. The view enables fine-grained control over the file system name space. Ioannidis *et al.* propose to implement privacy support as a system service [43]. The privacy support is a general data scrubbing service. Presumably, the service inspects files and network streams to scrub out sensitive information. In the system HiStar by Zeldovich *et al.* [95], information flows are explicitly controlled by the operating system. Instead of inspecting data streams, HiStar controls the permission of information flow between operating system components. Thus, with properly designed permissions, applications will not be able to leak sensitive information.

1.1.3 Privacy research in networking

Protecting privacy in an increasingly connected world is difficult. On the system level, research on privacy in networking is reflected in various works [10, 4, 52, 65, 35, 41] as privacy in ubiquitous/pervasive computing [91, 75, 74]. A basic mechanism discussed in these system-level privacy aware design principle is feedback and control of personal information disclosure, such as the six principals (notice, choice and consent, proximity and locality, anonymity and pseudonymity, security, and access and recourse) by Langheinrich [52].

Information sharing happens all the time, even in the years before the modern industrial revolution. It would be cumbersome and even impossible to report the disclosure of every piece of information. It even contradicts our common sense in real life. To this end, a fundamental question is the granularity of privacy enforcement. Adams [4] uses information sensitivity, information re-

ceiver, and information usage as the factors influencing people's perceptions of privacy. Palen and Dourish [65] suggest genres of disclosure as patterns for managing interpersonal privacy. Grudin and Horvitz [35] classify privacy control as three types: pessimistic, mixed, and optimistic, close to the three strategies by Jiang *et al.*: prevention, avoidance, and detection [45].

Recently, researchers have been looking into the implication of the increasingly frequent usage of different networks to privacy, especially widely deployed wireless networks. The issues that have been addressed range broadly across different layers of the ISO/OSI model. In the ISO/OSI model, lower layers provide general services to upper layers. Upper layer data is encapsulated in the lower layer messages, leaving lower layers to interact with other network entities. A big concern is that private information may be disclosed in the process of interacting with unknown network entities. Among all the topics, a large portion of work addresses location privacy in networks due to the commonly perceived threat of surveillance. In 2003, Kassinen identified privacy as one of the five themes of the user needs in location-aware services [46].

1.2 Location Privacy

Location privacy is addressed under two scenarios: explicit location information and implicit location information. Explicit location information is disclosed in location-based services. Implicit location information is disclosed in various mobile networks.

1.2.1 Location Privacy in Location-based Services

Location-based services (LBSs) are the kind of services utilizing locations as contexts. For example, a web search engine returns all the restaurants that are close to a mobile user's current location. Privacy in LBSs was an issue even at the dawn of pervasive computing. In 1996, Richard Harper mentioned the following comments by a colleague regarding the Active Badges developed at XEROX PARC in the earlier 90s [39].

> Do I wear badges? No way. I am completely against wearing badges. I don't want management to know where I am. No. I think the people who made them should be taken out and shot... [I]t is stupid to think that they should research badges because

> it is technologically interesting. They (badges) will be used to track me around. They will be used to track me around in my private life. They make me furious.

Under the control principle of privacy, location privacy can be preserved if the user has full control over his location information. This requires a client-centric approach to manage the user location information. The Cricket system [70] by MIT takes user location privacy as one of its design goals. A user merely receives location information from a decentralized network of beacons with ultrasonic waves, thus avoiding disclosing his location. Similarly, Smailagic and Kogan use a client-centric approach in their algorithm CMU-TMI to protect user privacy [81].

However, for those services that require explicit location information from the client, the above approaches can not be applied. In order to protect user privacy, location information needs to be obfuscated. This is a common practice. For example, traditionally civilian GPS signal has much lower resolution than a military GPS signal. Another example is intentional offset of strategic locations on a published map. Gruteser and Grunwald proposed location anonymity for the first time in 2003 [36]. Their approach is to lower the spatial resolution of the location information to encompass a group of users so that any specific user cannot be identified. Similarly, Gedik and Liu extended the location anonymity model by allowing different levels of anonymity to be provided to different users under the same framework [32].

The above works consider location privacy for a single location sample. In a mobile network, a user may need location-based services at multiple locations. The trajectory that has been taken by a user may be reconstructed if the sample is frequent enough. To protect the trajectory information, Hoh and Gruteser applied a perturbation approach for location anonymity. When two or more users come closer, their location samples will be slightly modified and perturbed so that a single path can not be reliably identified [40].

1.2.2 Location Privacy in Mobile Networks

Location privacy in mobile networks deals with implicit location privacy. Normally, when users move in a mobile network, various information can later be used as identity markers. For example, a MAC address in a wireless network is unique and can be resolved with the device later. Authen-

tication information is normally a unique user name, a unique password, or a unique certificate. Using those information sources at locations in a mobile network will disclose a user's trajectory.

The solution to this problem is to use multiple identities. Providing multiple identity-related information is addressed on multiple layers.

Physical Layer

From the first look of it, there appears to be no privacy issue on the physical layer since devices are designed without any personal information. However, radio fingerprinting has been used by mobile telephony carriers to discover cloned handsets. The same principle can be applied to track a single device using different wireless technologies. Finding a method to prevent tracking through physical layer fingerprinting remains a open problem. As Avoine and Oechslin [8] mentioned in 2005 regarding RFID (Radio Frequency Identification),

> Preventing traceability through radio fingerprinting seems quite difficult. There is no benefit for the manufacturers in producing tags that use exactly the same technology, producing the same radio fingerprint. Much more likely, manufacturers will experiment with different technologies in order to produce tags that have either better performance, price or size.

MAC Layer

In various mobile networks, each client device has a unique MAC address, such as Ethernet address in 802.11 networks, IMEI (International Mobile Equipment Identity) in GSM networks, and ESN (Electronic Serial Number) in CDMA networks. From the perspective of mobile telephony carriers, location privacy is used mainly to prevent any outsider other than the carriers themselves from learning the locations of a device. Measures have already been deployed by mobile telephony carriers. For example, in GSM networks, a TMSI (Temporary Mobile Subscriber Identification) will be assigned to a roaming device. For this reason, research has been focusing on emerging wireless networks, such as 802.11-based networks, wireless sensor networks, and vehicular networks.

Gruteser and Grunwald propose to use disposable/short-lived interface identifiers in a 802.11

network [37]. Hu and Wang also propose to change both MAC and IP addresses frequently [42].

However, despite those efforts, Pang *et al.* have recently shown that a lot more information other than MAC address can be used as an implicit identifier through fingerprinting [66]. For example, SSID probes, broadcast packet sizes, MAC protocol fields, and network destinations when no link layer encryption is used, are shown to be good implicit identifiers.

Although current privacy protection on the MAC layer is not enough, it needs to be pointed out that measures can be taken progressively to remove the above implicit identifiers. For example, a recent patch from Microsoft allows users to disable active SSID probing [57]. For wireless clients to negotiate services with access points privately, limited information exposure [96] can be employed.

Access Control

Access control is widely used in mobile networks to prevent unauthorized access. A user is assigned a unique secret key or digital certificate. The key or certificate will later be used to access the network. Apparently, the unique key can be used as a means to track the user because no other person can possibly possess this key.

Beresford and Stajano use pseudonyms for authentication and address the problem of changing pseudonyms without links between past pseudonyms and the present pseudonym [12]. Their solution is to let multiple users change their pseudonyms within a mix zone together so that the connection between past and present pseudonyms is anonymized. Hu and Wang propose to use blinded signatures as identities [42]. Gruteser and Grunwald use a modified EAP-TLS protocol to provide port authentication [37].

Network Layer

A simple example of location privacy on the network layer can be illustrated with a TCP/IP network. The geographical location of a IP address can be approximated with limited precision. Thus the location of a source can be determined based on its IP address.

The general approach to protecting location privacy is to rely on an intermediate trusted entity. For example, online anonymizers act as proxies so that people can surf the Internet anonymously.

In the mobile IP standard, the location privacy of a roaming user is protected through the use of its Home Agent. Similarly, Hu and Wang propose an anonymous bulletin board as an intermediate stop for data communication between a source and a destination [42].

1.3 Proposed Research on Location Privacy

We propose to investigate location privacy in two networks: wireless sensor networks and vehicular networks. Both networks are emerging networks and pose unique challenges to location privacy.

1.3.1 Application Background

A wireless sensor network is proposed to collect information about the environment in an economical and effective way by deploying small smart sensors into the environment. The deployment of sensors normally requires sufficient coverage for a useful application. The deployed sensors sense their immediate environment and send the results back to a base station for long term storage and further processing.

The deployment of sensor networks in the environment poses a big challenge to privacy as there is no physical protection of sensor nodes or there is a comparably high cost to protect sensors with respect to the cost of sensors. An eavesdropper can silently perform traffic analysis for days without raising any alarm.

The sensed packet has implicit location information associated with it. For example, a surge of packets within a certain area might indicate the occurrence of certain events. This could lead to the exposure of source location of those events, especially if the function of a sensor network is well-known. For example, in the panda-hunter problem [63], the network is well-known for monitoring panda activities. A poacher can increase his chance of finding a panda by moving to the source area of packets.

Vehicular networks grow out of the US Department of Transportation's Intelligent Vehicle Initiative (IVI) and Vehicle Infrastructure Integration (VII). IVI's vision is that vehicles will become smarter and smarter as computer and information technology is increasingly being applied. VII takes a broader view by requiring both intelligent infrastructure support and intelligent vehicles

and universal communication between vehicles, and between vehicles and infrastructure.

However, the information exchange that is required in IVI and VII presents a privacy risk if privacy is not considered as a design goal. Especially for location privacy, the privacy risk is perceived as surveillance if it is disclosed to "Big Brother", as physically endangering if it is disclosed to a malicious person.

For both networks, because they are deployed in a physical environment, they are susceptible to traffic analysis. Thus, we focus on preventing traffic analysis in order to protect location privacy. Danezis and Clayton gave general coverage on traffic analysis in their chapter "Introducing Traffic Analysis" [3]. Traffic analysis has military roots dated back to the First World War. Over the years, it has gradually been adopted in civilian cases. For example, some cannabis farms grow plants inside a house and require strong artificial sunlight. Police in UK searches for cannabis farms by looking for unusual heavy usage of electricity compared with past periods [3].

Location privacy is particularly vulnerable to attacks through traffic analysis. The "reality mining" project at MIT collects location and call information from 100 participants through a specialized software running on a cell phone [3]. The results show that calling or being with someone at 8pm on a Saturday night is a very good indicator of friendship. Furthermore, members of staff have much more predictable habits than students. It is thus possible to predict an individual's next location with high probability, given his past locations. Similar research done by Intel Research at Cambridge even accidentally revealed a undisclosed relationship between two members of the staff because the devices worn by them revealed they were meeting each other rather often at night.

1.3.2 Mechanisms to Protect Location Privacy

Fundamental countermeasures to traffic analysis include evading detection and camouflaging. Two modern wireless technologies, frequency hopping and code-division multiple access (CDMA), were originally designed for the military to reduce the chance of communication being detected by enemies. They are good examples of evading detection. In another military setting, to avoid a future military action being detected by suddenly increasing traffic between units, military communication links transmit a constant stream of encrypted messages, with dummy traffic being mixed with real

messages, to hide any traffic variation.

In civilian cases, preventing traffic analysis has mainly been addressed in anonymous communication over the Internet [13, 24, 59, 25]. The main purpose of anonymous communication over the Internet is to protect freedom of speech without fear of retaliation from a powerful adversary, such as a totalitarian government. In order to hide the association of communicating parties, messages are routed through a mix network. The mix network reorders messages from senders, presenting them to the receivers in a fixed order. Against traffic analysis, the effectiveness of a mix network depends on the number of communicating pairs within a useful time window. Essentially, any single association is hiding within this group of associations.

Applying anonymity in the general society context other than the Internet sounds controversial at first glance. Anonymity is the state of performing certain activities without revealing identity. A common attitude that shows the controversy of anonymity can be illustrated with the following statement, "What are you hiding, if you have done nothing wrong?" However, the necessity of anonymity in the general society context is best described by the conclusion of research conducted by National Academy of Sciences Study Committee on Authentication Technologies and their Privacy Implications [48],

> Preserving the ability of citizens to interact anonymously with other citizens, with business, and with the government is important because it avoids the unnecessary accumulation of identification data that could deter free speech and inhibit legitimate access to public records.

This dissertation is organized in the following way. In Chapter 2, we present protecting location privacy through evading detection with an application to sensor networks. In Chapter 3, we illustrate how to apply anonymity in protecting location privacy with an application to vehicular networks. Finally, chapter 4 presents using private information retrieval (PIR) to protect location information, with an application to navigation.

Chapter 2

Traffic Analysis

Traffic analysis is the technique of discovering information by observing the pattern of the traffic without checking the content of the traffic. Within the computer science field, this normally refers to analyzing an information communication pattern without the need to know the actual information being communicated. A common strategy in protecting privacy is by encrypting the information. However, under most circumstances, information communication itself cannot be concealed, which makes traffic analysis an effective attack. In this chapter, we illustrate such attacks in the context of energy-efficient wireless sensor networks.

To conceal traffic patterns, a general strategy is to hide the traffic pattern within other traffics. For example, a mix network [13] is widely proposed to conceal Internet access. This works well in scenarios where traffic is aggregated, for example, when the mix network is carrying multiple traffic flows. In a single-purposed network, such as a wireless sensor network, this approach no longer applies as there is no other traffic which can be used for concealing the traffic pattern. In this chapter, we explore alternative approaches that would apply in a wireless sensor network.

We first give a brief introduction of the appplication scenario. Then we illustrate threats against a wireless sensor network. Finally we present our proposal for this problem.

2.1 Introduction

One major application of wireless sensor networks is environmental monitoring. Sensors monitor their immediate environment and send events to a base station when predefined signals have been detected. In a multi-hop wireless sensor network, the events will be forwarded along a path to the base station. However, the transmission of packets on this forwarding path can be observed, leading to the source location of the events being leaked.

In a wireless sensor network, location information often means the physical location of the event, which is crucial given some applications of wireless sensor networks. For example, in a battlefield, the location of a soldier should not be exposed if he initiates a broadcast query. In the *panda-hunter* problem, the location of the panda should not be exposed to poachers [63].

An environmental monitoring network is normally a low duty cycle network. Often, traffic has a strong correlation with a certain event at a certain time. This gives big advantages to an eavesdropper since he does not need sophisticated techniques to discriminate traffic among different events. In this chapter, we study the source location privacy problem under the assumption of a single source during a specific period. Note that such a scenario can happen in a real wireless sensor network. In fact, this would be a typical case.

In the following sections, I first present the result that the source location information is leaked if there is only limited protection to the information. Then I present a mechanism to evade detection by a power-limited adversary.

2.2 An Example Attack against Flooding-based Phantom Routing

The dominating communication pattern in environmental monitoring sensor networks is from sensors to base station. Thus routing protocols in environmental monitoring sensor networks normally just set up a star-like topology for transmitting event packets from sensors to the base station. For a fixed base station, the topology is relatively stable. Thus, an attacker can backtrack to the source location if there are periodic recurrent event reports from the source location.

Phantom routing is proposed in [63, 47] to protect source location privacy. The basic idea

of phantom routing is to unicast each packet from the event source to a random intermediate location and forward it to the base station from that location. There are two varieties of phantom routing. One is called flooding-based phantom routing, in which the packet is delivered to the base station through flooding. The other is called unicast-based phantom routing, in which the packet is delivered to the base station through a single routing path.

In this section, we illustrate a simulated attack against the flooding-based phantom routing. We assume that the eavesdropper has minimum physical capability, which is the ability to detect the presence of a radio transmission. Also, to get a good estimate of the source location, the eavesdropper consists of a group of devices distributed in the network. Each device at a different location is considered an observation point. Although the number of observations is limited, we will show that the source location can still be approximated without much effort.

At each observation point, the eavesdropper can record the time of a radio packet. The propagation speed can be modeled as a Gaussian distribution and is unknown. Also, the time when the algorithm begins to flood a packet is unknown. So, the parameters to be estimated comprise the following tuple: (x, y, v, t), where (x, y) are the coordinates of the location where flooding begins, v is the propagation speed, and t is the time when flooding begins. Suppose that the coordinates of each observation point are (x_i, y_i) and the packet is observed at time t_i. The true distance between an observation point and the flooding source is:

$$D_i = \sqrt{(x_i - x)^2 + (y_i - y)^2} \tag{2.1}$$

The distance can also be written as:

$$VD_i = v(t_i - t) \tag{2.2}$$

Ideally, at each observation point we have $D_i = VD_i$. However, to estimate those four parameters, multiple observations at different locations are needed to solve the equation. Due to noise, the estimates at each observation may be inconsistent. To find the optimal solution, we use the mean

square error approach. We minimize the following formula:

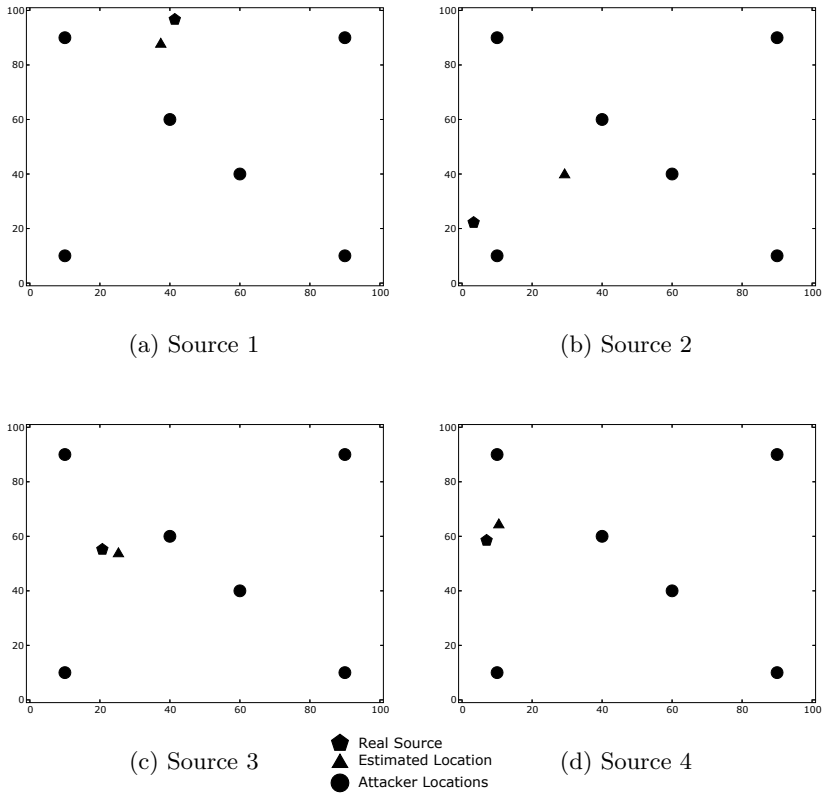
$$\sum |D_i - VD_i| \tag{2.3}$$

Ideally, four observation points should be enough for this purpose. However, in the simulation, we found that using six observation points yields much better estimates. Using six observation points compared with using four observation points requires only a modest increase in the tracking effort of an adversary. So, we present the simulation results with six observation points only.

To illustrate this attack, we have implemented the flooding-based phantom routing algorithm with TOSSIM [53]. We vary the number of hops during the random walk phase to check how this parameter affects the attack. The attack is being run over a network of 5000 nodes. We chose a large network size to show that even a large network can be susceptible to this attack. It's harder to preserve source location privacy in a small network under the assumption of only one single traffic flow existing in the network during a specific period.

We define the estimation error as the distance between the estimated location and the true location. To measure the effectiveness of the attack, we fixed attackers at six locations in the network and varied the location of the source. The simulated network spans a rectangular area of size 100×100. The communication range of every sensor is 2.25 units. Since the adversary is power-limited, this also means that the detection range of is only 2.25 units. The six locations of attackers are $(10, 90)$, $(10, 10)$, $(90, 10)$, $(90, 90)$, $(40, 60)$, and $(60, 40)$. The choice of the six locations is rather arbitrary provided that they are relatively far from each other and have good coverage of the network. Note that the chosen locations are not necessarily close to the real source. Figure 2.1 shows the estimation errors for different scenarios within a period in which 50 source packets were sent out. Table 1 shows the estimation errors and the summations of mean square errors.

We deliberately return very large cost values for unreasonable solutions so that the optimization can converge faster. For example, scenario 2 in Table 1 has a large cost value. The reason is that the real location is outside the convex set of the observation points while the optimization is trying to find some point within this convex set. We adopt the following strategy to overcome



(a) Source 1 (b) Source 2

(c) Source 3 (d) Source 4

Figure 2.1: Estimation Results for Four Sources

this limitation. An inaccurate estimate has a very large cost value, which can be used by the eavesdropper to trigger the movement of the observation points. To illustrate this strategy, we move the center of the original observation points toward the estimated location and re-estimate the location. However, during the moving process, if some observation points would move outside the network, we keep them at the boundary of the network. The whole process can be repeated. We use this strategy for the above example and the result is shown in Figure 2.2.

To investigate the effectiveness of the attack given different random walk steps, we vary the

Table 1. Estimation errors and Mean-square errors

Scenario	Estimation Error	Mean-square Error
1	10.0	2345.6
2	31.3	2.9×10^{34}
3	4.9	1588.2
4	6.7	2319.7

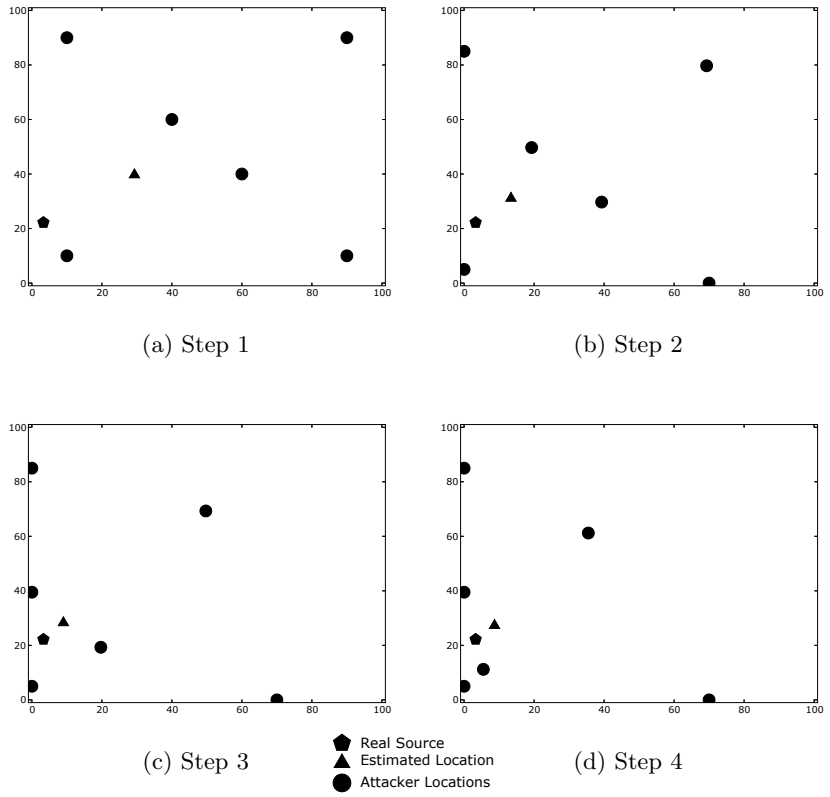


Figure 2.2: Strategy to Close in on Source 2

length of the random walk. During the simulation, we found that there are many local minimums in the topology we used above, where a node inside the network does not have any neighbor in one direction. This causes many packets to be dropped before reaching the flooding phase and deteriorates the estimate quickly. However, there is no suggestion on dealing with this problem in the phantom routing algorithm. To avoid the local minimum problem, we run the simulation on a network with 5000 sensors. The sensors are uniformly distributed in a 100×100 rectangle area. The increased density makes the local minimum a rare case.

Without loss of generality, we chose a source at $(25.0, 70.4)$. The random forward hop count is chosen for the values 5, 10, 15, 20, 25, and 30. The simulation results are shown in Figure 2.3. The estimation errors are larger for higher hop count values. However, even for a large hop count of 25, the estimate is still usable. Part of the reason that the estimation error gets worse is the way the phantom routing is designed. In our implementation, the forward directions are categorized

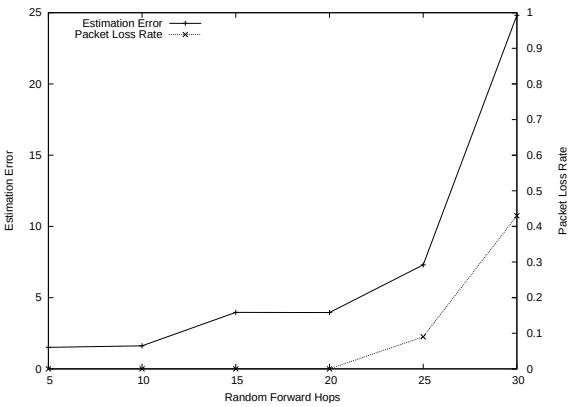


Figure 2.3: Estimation error for different random walk hop counts

according to sensors' x coordinates. For random forward hop counts of 25 and 30, some of the packets are forwarded to the boundary of the network and dropped since there is no recovery mechanism defined. In Figure 2.3, this is shown as an increased packet loss rate. Since the source is located closer to one side of the network, only packets being forwarded to the closer side are lost. This causes the estimate to move toward the other side of the network. For those hop count values without packet loss, the increase in estimation error grows only linearly with the hop count and the growing speed is much slower than that of the hop count value. This shows that varying only the random forward hop count is not effective for providing better source location privacy.

2.2.1 Drawbacks of Flooding

Privacy is lost when the adversary is able to predict the source location within a reasonable period of time. In the above illustrated attack, the adversary can predict the approximate position of the source when a single packet is flooded. Although randomness is introduced through the random walk phase, the adversary can improve the prediction through statistical estimation.

Modeling the routing as a random process, the effectiveness of the adversary's strategy depends on how randomness is introduced and on how the adversary can sample this process. Given a known random process, every sample contributes to the adversary's estimation of the invariant parameters. In our case, the parameters are the x and y coordinates of the source. To deter the adversary from predicting the exact location of the source, we would like to slow down the speed

at which the adversary can sample this process.

Assume that the source sends multiple packets to the sink over a period of time and uses consecutive sequence numbers to label those packets. The interval of packets received by the eavesdropper is defined as:

$$T = S_i - S_{i-1}, \tag{2.4}$$

where S_i is the sequence number of the ith packet from the source arriving at the same physical location. T is a random variable. The larger T's mean, the longer it takes for the adversary to get an adequate estimate of the source location. Note that the sequence number is used only for analysis. The packet does not have to have a sequence number.

Flooding is the worst method for protecting source location privacy in terms of T, which will take a fixed minimum value of 1 for all the locations in the network. Flooding enables the eavesdropper to accumulate information about the source location very quickly.

2.3 Greedy Random Walk

2.3.1 Random Walk and Source Privacy

The use of random walk is desired for protecting source location privacy. A random walk does not disclose any information about the source since the forwarding decision is made locally and independent of the source location. In fact, an eavesdropper can not distinguish two random walks from two different sources. Using random walk also forces the adversary to use backtracking strategies, rendering the attack described in section 3 impossible.

However, a pure random walk tends to stay around the source [47]. Define the hitting time T_a as the time when the Brownian motion path hits point a for the first time. Here a is any point other than the source. We have the following properties: [73]

$$P\{T_a < \infty\} = 1 \tag{2.5}$$

$$E[T_a] = \infty \tag{2.6}$$

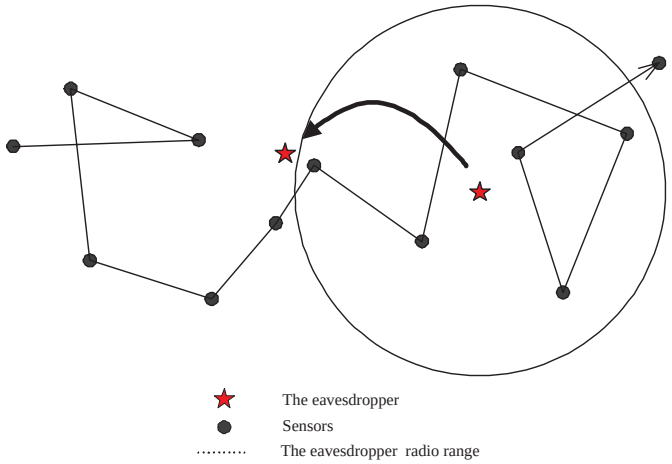


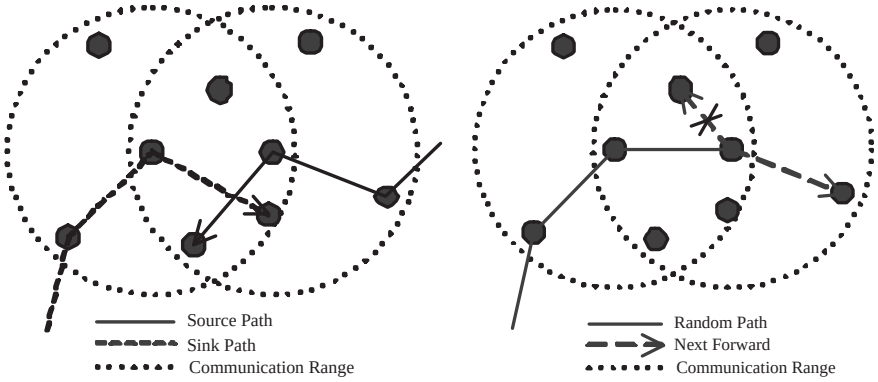
Figure 2.4: The eavesdropper beats random walk within its radio range

If we put the source at 0 and the sink at a, it means that although the Brownian motion path will hit a eventually, the average time it takes goes to infinity. It is not a desirable result since it means average unbounded delivery time.

Since a Brownian motion path eventually hits a, it is important to see at what speed it converges to a. Recent work from Shakkottai [78] investigated this problem. The convergence is quantified as how fast the non-delivery probability decreases. It is shown that the probability decays as $(log(t))^{-1}$, where t is how long the Brownian motion path has lasted. A more interesting result shows that if there is also a random walk from the sink at the same time, the probability of those two random walks not intersecting with each other decays as $t^{-5/8}$, which means that it is exponentially better than using only one random walk.

However, directly applying this approach is still not appropriate for practical applications because using random walk within the radio range of the eavesdropper is not useful to protect the source location privacy. For example, in Figure 2.4, the eavesdropper can move to the sensor from which it first hears the packet. Thus, the local random walk within the eavesdropper's radio range consumes extra energy and causes longer delivery time.

We propose a Greedy Random Walk (GROW) approach to address the above problems. In GROW, each time the sensor will pick up one of its neighbors that has not participated in the random walk. This way, the random walk is always trying to cover an unvisited area using a greedy



(a) Two Path Intersecting (b) Prevent Local Backtracking

Figure 2.5: Non-planarity in Communication Graph

strategy. Also, we eliminate local random walk and let both the source and sink initialize such a random walk to further improve the performance. The implementation of GROW is discussed in the next subsection.

2.3.2 GROW Algorithm

Previous analysis of random walk is based on a planar graph. However, this is not the actual communication graph in a wireless sensor network. If we treat the communication graph as a nonplanar graph during the implementation of the random walk, the probability of the source path and the sink path intersecting is much less than the previous asymptotic result. The scenario is shown in Figure 2.5a. We use local broadcasting to solve this problem. Whenever a sensor forwards a packet, all its neighbors overhear this packet and create a route entry for the source pointing to the forwarding sensor. This does not require additional transmissions. Essentially the random walk is sticky not only for the sensors on the forwarding path but also for the neighboring sensors of this path. In effect, we build a pipe along the forwarding path.

The scenario not only exists between two paths, but also exists on a single random path itself. A random path might backtrack to itself after some time. However, we would like the path to extend as far as possible and as quickly as possible. In Figure 2.5b, the sensor might forward the packet to one of its previous hop's neighbors. Such a forwarding decision is not good since the random walk does not make much progress. To prevent this case, we use a Bloom filter [16] to

store all current neighbors in the forwarding packet. When the next hop randomly picks up one of its neighbors, it checks whether that neighbor is already in the filter. Given a limited number of neighbors, the probability of false positives can be made very small by using a reasonable sized filter within a packet. In other words, the packet will be forwarded to a sensor that has not seen the packet before with high probability.

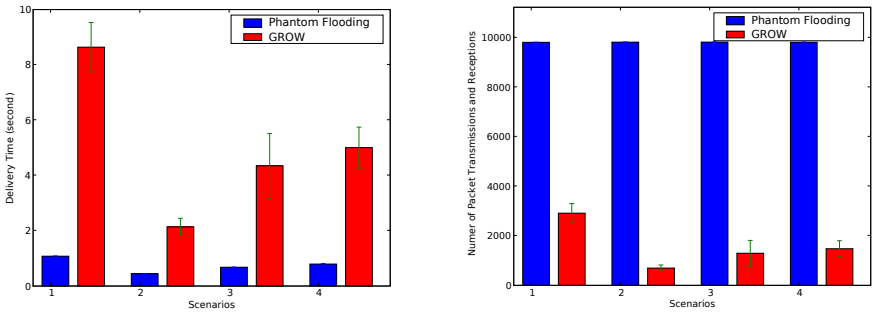
To decrease the chance of backtracking, each sensor keeps a Bloom filter to store those neighbors that have already participated in the forwarding. Each time a sensor is forwarding a packet, it will store the last hop from which the packet came and the next hop which it forwards the packet to. When the random walk backtracks to a sensor, it will choose one neighbor that has never forwarded the packet before. In this way, we hope to maximize the coverage given a fixed path length.

If the source and the sink are close to each other, the two random paths have a greater chance to intersect, thus the intersection points are closer to the source and the sink. This enables the eavesdropper to possibly trace the path. To prevent this from happening, we require a minimum path length of the source random walk.

Note that we do not assume any routing infrastructure in GROW for generality. If extra information is available, we can certainly use the information to improve the performance. For example, if the geographical locations of sensors are known, it is easy to identify which part of the network has not been visited. Thus, a more effective greedy forwarding based on this information can be used.

2.4 Performance Evaluation and Analysis

We implemented our algorithm in TOSSIM [53]. For comparison, the simulation is run over the same topology we used in Section 3. The topology is generated through uniformly deploying 5000 sensors within a rectangular area of 100×100. The communication range for each sensor is 2.25 units. We have tested our algorithm on several topologies; however, the algorithm does not perform significantly different on topologies generated with different random seeds. Thus, we present simulation results over only one topology.



(a) (b)

Figure 2.6: Comparison between flooding-based phantom routing and Random Walk for Different Scenarios

2.4.1 Delivery Time

Ideally, the shortest delivery time is achieved if the packet is forwarded along the shortest path from the source to the sink. In flooding, the packet is forwarded approximately on this path since the propagation follows a wave pattern. This approximation can be proved through induction.

To provide source location privacy, it is necessary to relax the requirement for the delivery time. This is because if packets are always forwarded through the shortest path, it is easy for the eavesdropper to backtrack the path. There is certainly a trade-off between privacy and delivery time. We compare the delivery time between the flooding-based phantom routing and GROW. In Section 2.4.3, we show that privacy protection is provided by GROW.

For comparison, we use the same set of sources as those in Figure 2.1 and the same sink for both the flooding-based phantom routing and GROW. For the flooding-based phantom routing, we chose a rather conservative random walk hop count 5. For GROW, we fix the minimum path length of the source random walk to be 50. This value is chosen based on the scale of the network diameter.

Figure 2.6a shows the delivery time for both the flooding-based phantom routing and GROW. The delivery time is measured in seconds. Figure 2.7 is the cumulative distribution of delivery times for scenario 1. Although in scenario 1 the average delivery time is increased from 1 second to 9 seconds, over 50 percent of the packets are delivered within 5 seconds and 80 percent of the

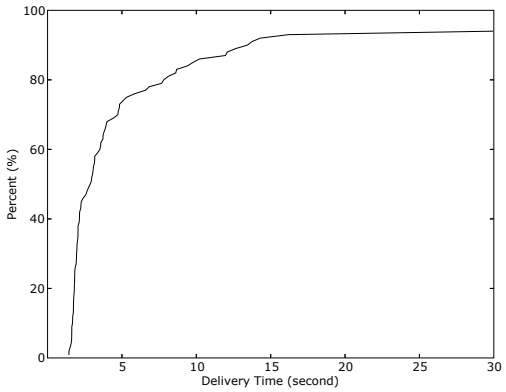


Figure 2.7: Cumulative Distribution of Delivery Time in Scenario 1

packets are delivered within 12 seconds. In other scenarios, the sources are closer to the sink. Thus, the delivery time is considerably less.

2.4.2 Energy Consumption

In wireless sensor networks, packet transmission is generally the most power consuming operation. Packet reception also consumes significant energy, often on the same magnitude as packet transmission. To simplify the analysis, we assume that one packet reception consumes the same energy as one packet transmission and omit other energy consumption aspects of delivering a packet. The comparison is done on the total number of packet transmissions and receptions.

Figure 2.6b shows the comparison between the flooding-based phantom routing and GROW. The flooding-based phantom routing consumes the same energy regardless of the source. GROW consumes much less energy. Even in scenario 1, where the packet exists in the network much longer than it does using in the flooding-based phantom routing, the energy consumed by GROW is still less than half the energy consumed by phantom routing. This shows the benefits of using random walks from both the source and sink. Although GROW tends to cover all the sensors, in practice it needs to cover only a small portion of the network to have the packet delivered.

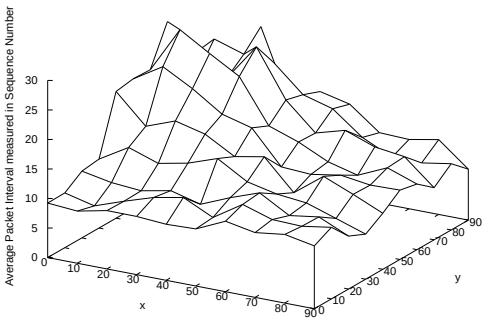


Figure 2.8: Average Interval of Hearing a Packet at the Intersection Points for Different Sources across the Network

2.4.3 Privacy Protection

In our approach, an eavesdropper needs to stay on the random path to track down the source. Since the random path from the sink is relatively stable, the best strategy for the eavesdropper is to start from the sink and backtrack the last hop each time he overhears a packet. However, to really get to the real source, the eavesdropper also has to backtrack the source random path. Since each packet from the source follows a different random path, the only relatively stable information which can be utilized by the eavesdropper is that the source path will intersect the sink path at some point. If the eavesdropper can not predict the next intersection point, he will miss the chance to make progress toward the source. To measure the privacy protection of our approach, we used the metric defined in Section 3 to calculate the mean interval between packets from the same source arriving at the same intersection point on the sink path.

To verify that we have consistent privacy for different sources across the network, we picked one hundred sources located approximately on a 10×10 grid over the network. The average intervals are shown in Figure 2.8. All average intervals are over 7 and the actual intervals are randomized. This makes it very hard for the eavesdropper to reliably catch a packet. Even if he stays at one location and eventually receives a packet, the last hop of the packet might be some location he has already visited since each time the packet follows a different random path. In such a case, he gets no new information and makes no progress toward the source.

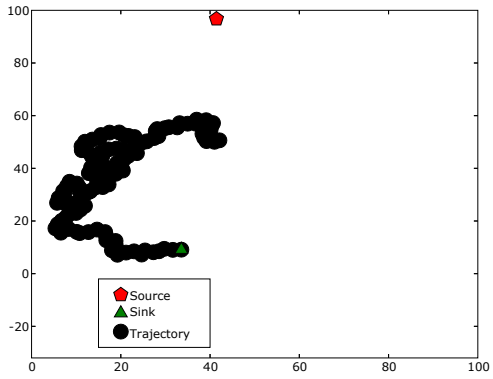


Figure 2.9: Trajectory of the Eavesdropper over 250 packets

To verify that our approach will not lead the eavesdropper to the source, we implemented a backtracking algorithm to simulate an eavesdropper. Originally, the eavesdropper stays near the sink. Once he detects a packet, he moves to the new location if he has never been there before. We run this algorithm over the simulated scenario 1. Figure 2.9 shows the trajectory of the eavesdropper during a period in which 250 source packets were sent out. The black dots are locations the eavesdropper has visited during this period. Although it looks like that the eavesdropper is approaching the source, the progress is slow. Ultimately the eavesdropper can reach the source since the greedy strategy eventually visits every sensor in the network. However, compared with flooding-based phantom routing, in which the eavesdropper can compute a very good estimate of the source location within a period of only 50 packets, the privacy protection is improved significantly.

Chapter 3

Anonymity

Identity information is one type of private information. In many scenarios, it is sufficient to remove identity information in order to protect privacy. This is especially needed when the actual information associated with the identity needs to be preserved for the specific applications. For example, medical records are sensitive private information. Often, the actual medical condition is needed for research purposes. A common strategy in protecting medical privacy is to remove the identity information associated with the medical records. This process is called anonymization. In this chapter, we apply novel anonymizing techniques to vehicular networks. We first describe our application scenario. Then, we explain in details our anonymization protocols.

3.1 Introduction

Each year, over six million crashes occur on U.S. highways, killing more than 42,000 people, injuring approximately three million others, and costing more than $230 billion per year. To reduce the number and severity of these crashes, the Intelligent Vehicle Initiative (IVI) was launched in 1998. Over the course of 8 years, numerous advanced vehicle safety technologies have grown out of this initiative [44]. For example, in a 2003 national IVI meeting, a dynamic warning light has been demonstrated over an intersection to prevent intersection collision if the laser sensor detects an incoming intersecting vehicle [30].

An emerging application is Vehicle Infrastructure Integration (VII) planned by the U.S. Depart-

ment of Transportation (USDOT) in the coming years [89]. Its goal is to provide a communication link among vehicles on the road, and between vehicles and road-side infrastructure to improve safety, efficiency, and convenience of the transportation system. In the U.S., Directed Short Range Communication (DSRC) is the standardized protocol for this communication link, working in the 5.9 GHz band allocated by the FCC.

In this chapter, we first give a brief analysis of privacy requirements in vehicular networks. Then we discribe the anonymization protocol based on public key cryptography. Followed by the analysis on how it would perform in the mobile scenario. We propose a graph-based metric called anonymity graph to measure it. At last, we show that the anonymity protocol can be similarly applied with symmetric cryptography.

3.2 Potential Applications

Applications that have been proposed for vehicular networks are roughly categorized into two classes. The first class is safety applications. The second class is traffic information applications. The US National Highway Traffic Safety Administration (NHTSA) and the Vehicle Safety Communications Consortium (VSCC) of CAMP (Crash Avoidance Metrics Partnership) have chosen the following eight high-priority vehicular safety applications [72]: Traffic Signal Violation Warning, Curve Speed Warning, Emergency Electronic Brake Lights, Pre-Crash Warning, Cooperative Forward Collision Warning, Left Turn Assistant, Lane Change Warning, and Stop Sign Movement Assistance.

In the literature, various safety applications have been addressed by the academic community, such as collision avoidance [28], intersection collision warning [58, 27], traffic merging [90], adaptive traffic light [34], etc.

Traffic information applications target improved traffic information dissemination. Examples include TrafficView [60], Self-Organizing Traffic Information System (SOTIS) [93], and parking space finder [18], etc.

3.3 Security Requirements

Both classes of applications exhibit unique characteristics on security and privacy. Safety applications require high-fidelity information being exchanged in real-time. The IEEE 1609.2 standard on secure message formats suggests the use of a Public Key Infrastructure [2]. However, as pointed out by Studer, Luk, and Perrig [84]: "identification of vehicles is of little importance to VANET safety applications. ... but location and movement of the signer needs to be verified." Thus location information needs to be as precise as possible.

However, the usage of this location is limited to the local neighborhood as safety applications rely heavily on single-hop broadcast communication with nearby vehicles and infrastructure [72]. Thus it is not necessary to expose location information over long distances. To protect location privacy, this location information should be used only within a short distance and time period. Studer *et al.* proposed an identity-free location verification scheme in [84]. Their protocol is an anonymous protocol since no identity is exposed.

Various traffic information is used to improve transportation system efficiency. Traditional traffic information systems use different sensors, such as inductive loop, fiber optical sensor line, Doppler Radar speed sensor, magnetic sensor, piezoelectric sensor, and traffic video camera. The information collected include vehicle count, vehicle speed, vehicle density, vehicle type, and vehicle weight. It should be noted that although there are initial efforts of investigating re-identification of vehicles [85, 86, 61], it is only used for travel time estimation in specific road segments and does not represent pervasive surveillance.

An alternative to in-road sensors is to use a portion of vehicles as "mobile probes" into traffic flows [83]. This portion of vehicles can be public transportation vehicles, patrol cars, etc. SOTIS [93] generalized this concept by requiring that such vehicles participate in a self-organizing ad-hoc network with Inter-Vehicle Communication. To efficiently distribute traffic information and form a consensus from the information collected, various aggregation techniques have been proposed in [33, 60, 93, 55], etc. To deal with cheaters, security mechanisms were proposed in [33, 71, 68] to perform validation and provide accountability.

To enable the pervasive deployment set in VII, it is desirable to alleviate users' privacy concerns

by protecting user anonymity. Although conditional anonymity have been the focus of recent research efforts [87, 54, 38, 17], we argue that unconditional anonymity is a useful concept in vehicular networks and can coexist with conditional anonymity.

Anonymity is the state of being unknown. In access to a computer system, it is often reduced to not being able to be distinguished among a group of individuals to enforce limited access control. In order to prove just the membership within a group, various cryptographic constructs can be used. A ring signature is a cryptographic primitive to provide unconditional anonymity for sending out information. A realization of ring signature in vehicular networks has been proposed by Gamage *et al.* for reporting accidents [31]. In a similar way, a realization of anonymous authentication proposed by Sha *et al.* enables the use of interactive protocols with the designated infrastructure [77]. In both schemes, vehicles dynamically share their public keys to construct anonymous proofs. Both schemes integrate with public key infrastructure seamlessly.

In the subsequent sections, we first illustrate our anonymous authentication protocol based on public keys. Then we analyze its effectiveness for protecting anonymity. Third, we present preliminary results on the effectiveness of a shared keys scheme on protecting anonymity.

3.4 Anonymous Authentication Protocol

Our anonymous authentication protocol is based on a cryptographic construct called *verifiable common secret encoding* [76]. Using *verifiable common secret encoding* enables us to provide adjustable anonymity.

3.4.1 Verifiable Common Secret Encoding

The *verifiable common secret encoding* is based on public key cryptography. For a group of n users, each with its public/private key pairs (Pub_i, Pri_i), the verifiable common secret encoding is constructed as $(Pub_1(x), Pub_2(x), Pub_3(x), ..., Pub_n(x))$. Any member i of the group can decrypt its corresponding encoded message $Pub_i(x)$ with its private key Pri_i. It encrypt x with the other $n-1$ public keys to obtain its own copy of the encrypted secrets $Pub_1(x')$, $Pub_2(x')$, ..., $Pub_n(x')$. It then verifies that these secrets match those that were received. If they all match up, it accepts

the secret value x and proceeds with the remaining protocol steps.

3.4.2 Key Management and Group Formation

Each vehicle possesses a pair of public/private keys. Then vehicles are grouped together. The group is initialized and managed in the way described below.

A central server manages group information. During group initialization, the server first estimates the needed number of the groups, N_g, based on the estimated number of the total potential vehicles, N_v, and the minimum and maximum numbers of vehicles in one group, n_1 and n_2. Thus, the N_g should satisfy $N_g = \frac{1}{2}(\frac{N_v}{n_1} + \frac{N_v}{n_2})$. For each group member, we build a pair of public/private keys, assign an index for that key pair, and maintain a group version. These members are kept as dummy members until they are assigned to new vehicles, which is usually done by the second level key distribution servers. If the key distribution server and the central server are not cooperating, although there are a lot of dummy keys in each group, the central server has no idea about which key is already issued. If they cooperate, we can delay the function of the whole system until all groups have enough keys distributed. After initialization, all the keys in the group are organized to a complete binary tree, whose breadth-first travel results in an ordered sequence of the corresponding key index. Each subtree root is assigned an ID, which will be used by any member of the subtree in the authentication process. Each vehicle stores IDs of all the subtrees it belongs to.

The dynamic group management is conducted in the following ways. When the keys are revoked, the previous member which holds the key is no longer valid. Thus, the central server will replace the invalid key using a new key pair and update the group version. When a new member joins, assuming the central server has the information about the number of dummy keys in each group, it will be put to a group with the most dummy keys, and the server will find the first dummy or empty slot in the key tree based on a breadth-first search and distribute the corresponding private key, the key index, the group version, and the public key of the whole group, to the newly joined member. The updated group information should be distributed to the authentication server and other members in the group, which is a challenge in a large distributed system such as vehicular networks. Fortunately, we can assume that membership updating is infrequent. For example,

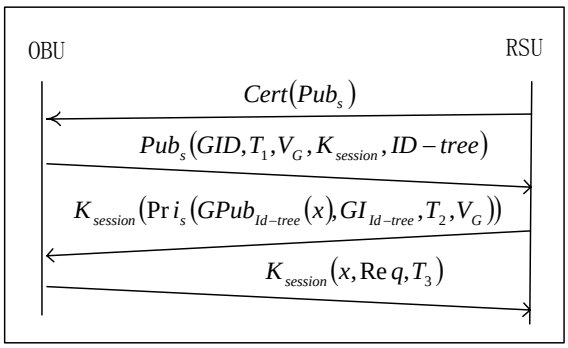


Figure 3.1: Group-based Authentication Protocol.

assume that a vehicle will be removed from a group if it is stolen. In the US, the average number of vehicles stolen in 2003 is 1 in 190. [1] For a group with 100 vehicles, there is about 50 percent chance that one vehicle in the group is stolen per year. This is a very small change for the group. When a group is changed, a push approach is used to update the group information to the caching servers. The updating of group information to the group member is integrated with the process of the authentication.

3.4.3 Protocol Description

The protocol is depicted in Figure 3.1. It has 4 steps:

1. RSU→OBU: $Cert(Pub_s)$.

   RSU announces its presence periodically with its certificate.

2. OBU→RSU: $Pub_s(GID, T_1, V_G, K_{session}, ID-Tree)$.

   OBU constructs a message with its group identifier GID, current time T_1, its current group version V_G, a session key $K_{session}$ selected by itself, and a subgroup identifier $ID-Tree$. It then encrypts the message with RSU's public key Pub_s so that only the RSU is able to decrypt it.

3. RSU→OBU: $K_{session}(GPub_{ID-Tree}(x), T_2, V_G, Pri_s(H))$.

   RSU constructs the verifiable common secret for the subgroup $ID-Tree$ of the group GID with a random value x. It then constructs the challenge message with the verifiable common

secret, its current time T_2, its current group version V_G, and the signature $Pri_s(H)$ obtained through encrypting the message hash H with its private key Pri_s.

4. OBU→OBU: $K_{session}(x, Req, T_3)$.

   OBU decrypts x from the verifiable common secret and verifies the anonymity guarantee. Upon successful decryption and verification, it constructs a reply message with x, service request Req, and its current time T_3. The message is encrypted with the session key $K_{session}$.

3.4.4 Adjustable Anonymity

Our protocol provides dynamically adjustable anonymity. Notice that the maximal level of anonymity is bounded by the group size. The adjustable anonymity achieves anonymity levels below this maximal anonymity.

The adjustable anonymity is achieved through tuning the two dominant factors within the protocol: communication cost and computation cost. The communication cost is tuned by changing the $ID-Tree$ used in the protocol. By changing it to a different one at a different level of the group tree, the subgroup size changes and the size of the challenge message changes. The computation cost is tuned by using probabilistic verification at OBUs. In probabilistic verification, instead of computing all the $n-1$ encrypted secrets $Pub_1(x')$, $Pub_2(x')$, ..., $Pub_n(x')$, OBU computes m ($m < n-1$) encrypted secrets randomly chosen out of the $n-1$ secrets and compares only these secrets with the m corresponding secrets that were received. Because the computation cost of public key encryption is relatively expensive, by changing m, the computation cost is changed.

3.4.5 Analysis of Probabilistic Verification

The probabilistic verification is illustrated with the following example. If the common secret set includes 50 encrypted values, and each encrypted value has a probability of 60% to be verified, as a result, about 30 encrypted values are verified on average, and a lot of computation cost is saved. However, probabilistic verification introduces a risk of the OBU suffering from a reduced level of anonymity. The RSU might use different random values to construct the 50 encrypted values. It then identifies the OBU based on the value returned from the OBU. We call the use of different

random values server probing. Next we analyze the effect of the probabilistic verification on the anonymity in cases both with and without server probing.

Without server probing, probabilistic verification will not affect the anonymity because the same value is encrypted in all common secrets. We analyze the anonymity in this case. In our protocol, the OBU may choose different sub-trees in each authentication. Thus, it is impossible for the RSU to link these trees to figure out the identity of the OBU by using spatial-temporal analysis. The RSU has to guess the ID of the OBU by chance. Then, the anonymity that can be expected is determined by the number of nodes in the tree, denoted as $|T|$ and bounded by the depth of the tree, d, where $2^d \leq |T| \leq 2^{d+1}$. The probability of successfully guessing the vehicle's identity is the multiplicative inverse of the number of nodes in the sub-tree. For instance, in the extreme case of only one member in the sub-tree, the RSU can easily know the person that is communicating with him. For a sub-tree with $|T|$ nodes, the RSU has only a $1/|T|$ probability to identify who is talking with him even when the authenticator knows which vehicles are in this group. The maximum anonymity can be expected in the case that the RSU is not actively probing the identity of the OBU.

In the case of the RSU trying to probe the OBU's identity, the anonymity of the OBU may be reduced. If the RSU wants to identify the OBU exactly, it has to use different numbers for different members. In such a case, the OBU can easily detect the probing by verifying only one other slot. So, let's assume that the RSU is just trying to decrease the anonymity level of the OBU by using the same number for a subgroup of s slots, which is not distinguishable to the RSU. In this case, if the OBU verifies only m slots where $m < s$, the probing may not be detected by the OBU. However, we will show that the probability of the probing not being detected, Pr, is very small.

Assume that there are $|T|$ members in the sub-tree, the RSU uses a subgroup of size s to probe the OBU's identity, and the OBU randomly verifies another m slots, in addition to the private key decryption for obtaining the challenge.

The selection of s slots by the RSU is independent of the selection of m slots verified by the OBU. So we can assume that the s slots are determined and m slots are randomly drawn out of

n slots. Among the s slots, one slot corresponds to the OBU. I.e., there are exactly $s-1$ slots that hold the same challenge value as that for the OBU. The probing is successful when the OBU chooses the m slots out of the $s-1$ slots. The total number of such cases is:

$$C^m_{s-1}$$

However, the total number of ways to choose the m slots out of $|T|-1$ slots is:

$$C^m_{|T|-1}$$

So the probability of probing not being detected is:

$$P = \frac{C^m_{s-1}}{C^m_{|T|-1}} \tag{3.1}$$

We define the anonymity reduction factor as the ratio of the size of the subgroup to the size of the original group. So the anonymity reduction factor r is defined as:

$$r = \frac{s}{|T|} \tag{3.2}$$

Now, we have:

$$\frac{s-i}{|T|-i} < \frac{s}{|T|} \text{ for } i > 0 \tag{3.3}$$

So,

$$Pr < \left(\frac{s}{|T|}\right)^m = r^m \tag{3.4}$$

This means that the probability of successful probing by the RSU decreases exponentially as the number of verifications done by the OBU increases. The rate of decrease is proportional to the anonymity reduction factor. So the RSU can not reduce the anonymity significantly. Otherwise, the probing by the RSU is easily detected. For example, let $m = 20$ and $Pr = 0.01$, then r must be greater than 0.79. In this case, if $|T|$ is 100, then the OBU can be confident that it can detect the reduction of the anonymity from 20 to 79 with probability 0.99 even if only 20 slots are verified.

The above analysis shows that if we use a probabilistic anonymity definition, there can be much flexibility in choosing the protocol parameters. So we define the anonymity here as a tuple, $< P, PL >$, where PL is the anonymity level and P is probability that any reduction to the anonymity level can be detected. We have shown that there is a mapping between the expected anonymity, which is $< P, PL >$, and the two parameters in the protocol, the number of the nodes in the tree, $|T|$, and the number of verifications done by the OBU, m. Thus, based on the two parameters we can estimate how much anonymity can be expected by the mobile user. And on the other hand, based on the expected anonymity requirements, the mobile user can set up the correlation between the two parameters. Based on above formula, we can deduce the relationship between these four parameters as listed below.

$$P = 1 - Pr > 1 - \left(\frac{PL}{|T|}\right)^m \tag{3.5}$$

where, the left part of the formula specifies the expected probability, P, of the expected anonymity level, $|PL|$. And the right part of the formula shows the probability of the expected anonymity level based on the calculation of the system parameters.

Given specific values of m and $|T|$, a high anonymity level can be achieved with low probability and low anonymity level can be achieved with high probability. We fixed the two parameters $|T|$ and m and draw the relationship between P and PL in Figure 3.2.

In Figure 3.2, the x-axis is the desired probability to detect the reduction while the y-axis is the corresponding maximal reduction the RSU can achieve. For a certain anonymity level, a further reduction can be achieved with even higher probability of being detected. However, we want to have a strong guarantee for the anonymity level we choose. So it is desired that the further reduction has only a very slim chance. In Figure 3.2, this is shown as the right part of a curve. For all values of m, it drops rapidly when P is greater than 0.9. Based on this, we suggest to use $P \geq 0.9$. As shown by the previous example, significant computation cost can still be saved for this choice of P.

On the other hand, if we have determined the value of P and PL, we can find the relationship between the $|T|$ and m,

$$\left(\frac{PL}{|T|}\right)^m \leq 1 - P \tag{3.6}$$

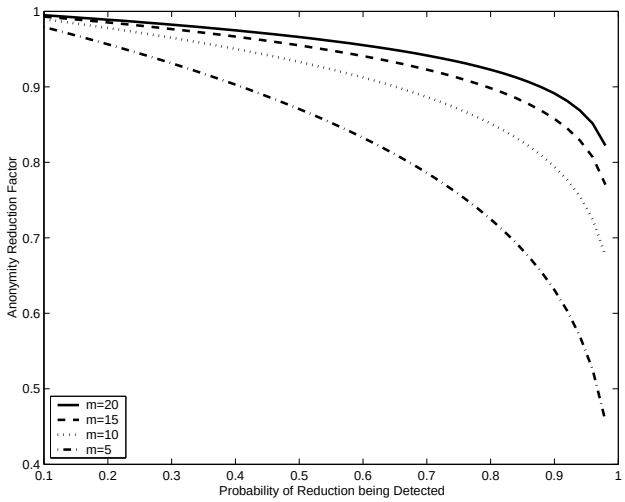


Figure 3.2: The Relationship between Probability of Detecting Reduction and Anonymity Reduction Factor.

For a certain anonymity level PL, and a probability P to detect any reduction, the OBU can either choose a large $|T|$ or a large m. The relationship is shown in Figure 3.3.

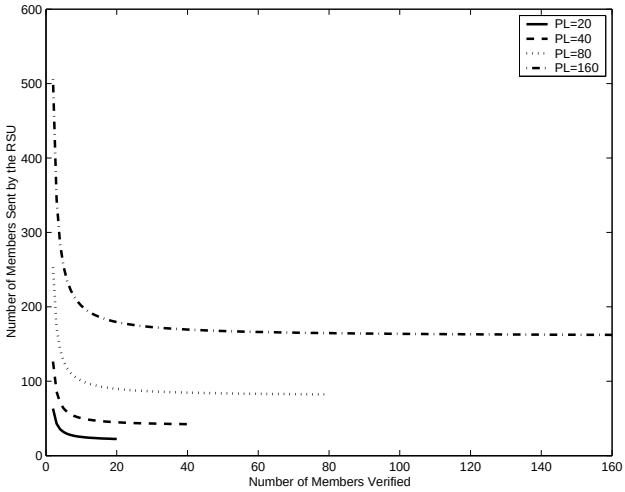


Figure 3.3: The Relationship between Number of Members Verified and Number of Members Sent. We use $P = 0.9$ for all cases.

In Figure 3.3, the x-axis is the number of members being verified while the y-axis is the corresponding total members that should be requested from the RSU. This figure shows that when the OBU verifies only 20 members, the number of members being sent approaches the minimum required value in all four cases. It also shows that the increased number of members being sent

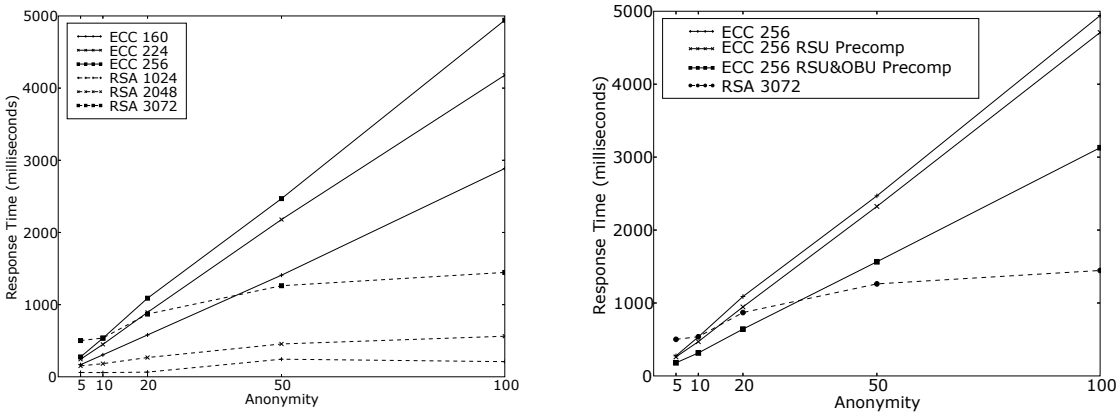
is exponential to the decreased number of members being verified for small values of m. So it is generally preferable to use a reasonably large m. It also shows the effectiveness of the probabilistic verification and discourages the RSU from reducing the anonymity level since the reduction can easily be detected by the vehicle with much lower cost.

The above analysis assumes that the vehicle requires a certain anonymity level and investigates the tradeoff between the communication cost and the computation cost. It is our conjecture that, to have both reduced communication cost and reduced computation cost, the overall anonymity level has to be reduced. The reduced anonymity level can be specified with a reduced P, a reduced PL, or both. Again, our analysis can be applied for flexible selection of the protocol parameters.

3.4.6 Performance Evaluation

We conducted simulations to evaluate the performance of the adaptive anonymous authentication protocol. We implemented the authentication protocol with the Crypto++ [23] package. Since the number of challenges is proportional to the size of the group, to minimize the communication delay, it is better to use public key cryptography with the most security per bit. For this purpose, we choose Elliptical Curve Cryptography (ECC). For comparison, we also evaluate the performance of the protocol based on the widely used RSA cryptography. The result is shown in Figure 3.4a, where the x-axis is the number of users (i.e., anonymity level) in a group.

The time spent on the authentication based on ECC is linear with the anonymity level. This is expected since in this case cryptography computation is dominant and scales with the number of members in the group. RSA-based authentication performs better than ECC-based schemes. The reason is that most computation involved in the authentication is public key encryption. ECC public key encryption is considerably slower than comparable RSA public key encryption with the same security strength. This is shown in the Crypto++ benchmarks [23]. However, RSA-based authentication generates much longer packets. To roughly estimate the communication cost, the number of DSRC packets required for transmitting the common secret for two different anonymity levels and different cryptographic key sizes are shown in Table 3.1. A 3072 bit RSA-based signature for a group with 100 members uses about 98 DRSC packets. So it is more likely to suffer from



(a) Response Time Comparison between ECC and RSA(b) ECC Performance Improvement through Precomputation

Figure 3.4: Response Time Evaluation.

Table 3.1: Size of the Common Secret in DSRC Packets for 100 Members.

Public Key Cryptography	Number of Packets for 100 Members	Number of Packets for 50 Members
ECC 160	21	10
ECC 224	30	15
ECC 256	34	17
RSA 1024	32	16
RSA 2048	64	32
RSA 3072	98	49

packet losses on wireless communication channels.

Figure 3.4b shows the performance improvement of 256 bit ECC through precomputation. For comparison, we evaluated two configurations of precomputations. One is to precompute at only the authentication server. The other is to precompute at both the server and the user. It can be clearly seen that the computation on the user side is the bottleneck of the protocol. As we showed previously, verifying only 20 slots is enough to provide a higher level of anonymity with high probability. Our probabilistic verification not only provides flexibility but also favors a vehicle-side implementation.

3.5 Effectiveness of Public Key-based Anonymous Authentication Protocol

The anonymous authentication protocol in the previous subsection provides computational anonymity in one round of authentication. In vehicular networks, it may be susceptible to attacks through traffic analysis. For example, the unique set of public keys used by a vehicle may enable the system to infer its traveling trajectory. The trajectory may expose its identity through its starting location or destination. This scenario is very similar to the attack against a stream of packets in anonymity systems.

A natural modification is to let a vehicle use different sets of public keys. Even so, its identity may still be narrowed down to the intersection of those sets, thus reducing its anonymity.

In this subsection, we present the achievable anonymity by the abovementioned unconditional anonymity schemes in vehicular networks. We look at two ways that vehicles share their public keys. We derive the necessary conditions to preserve anonymity and show by simulation that without any strong agreement among vehicles, anonymity can still be very well preserved given properly chosen parameters at each vehicle. To facilitate discussion, we propose to use *mobile anonymity* to denote the achievable anonymity in this "streaming" case. Note that *mobile anonymity* is not computational anonymity. It is rather a system-level anonymity.

3.5.1 Mobile Anonymity

In dynamically constructing a group, different members of the group may not have a consistent view of the group membership. This is due to two reasons. One reason is that the use of group members at each vehicle is decided by each vehicle. Different vehicles may use different group members. The other reason is that different vehicles may hear a new vehicle at different times. This issue is not specific to vehicular networks. It applies to general mobile scenarios in which different users come and go. Thus we term the anonymity in a mobile scenario *Mobile Anonymity*.

Mobile Anonymity is significantly different from static anonymity. First, the group changes much faster than a static anonymity group. Second, the group size is often limited due to the

limited range of choices for group members.

In order to cope with the inconsistency among group members, we use a graph model to check its effect on the overall anonymity. We define an *anonymity graph* for a dynamic group as the following.

Definition 1. An anonymity graph $G = (V, E)$ is a directed graph.

1. V is the vertex set. Each vertex corresponds to a vehicle that is participating in the group.

2. The edges in E are constructed in the following way. For each vehicle V_i, if it uses V_j as its group member, there is a directed edge from V_j to V_i in E.

3. Each vertex V_i has a *true identity* which is represented with V_i. Each vertex V_i is also marked with a sequence of identities that can be assigned to it.

Note that an anonymity graph is only used as an analysis tool. Each vertex in G has a true identity that corresponds to the vehicle it represents. To an outside observer, the true identities of vertices and the edges are not known. An outside observer can only deduce the identity of a vertex from the set of identities it uses.

In this section, we generalize the concept of *Mobile Anonymity* under an ideal model in which each vehicle sends exactly one request. In Section 3.5.2, we apply this model to vehicular networks by considering some real issues. For example, one vehicle cannot be at two locations at the same time. Thus two requests at two different locations at almost the same time are clearly from two vehicles.

Under this graph model, a static group is mapped to a complete graph since each vehicle will be used by all the other vehicles. It provides maximum anonymity where every identity is used by every other vehicle. Clearly, anonymities for different members are closely related with each other. In the following, we define *group anonymity* to characterize this relationship among group members in a dynamic group.

Definition 2. The group anonymity $\mathcal{K}(G)$ of an anonymity graph G is defined by the number of possible ways to assign identities to vehicles. A valid assignment $\mathcal{A}(G)$ has the following properties:

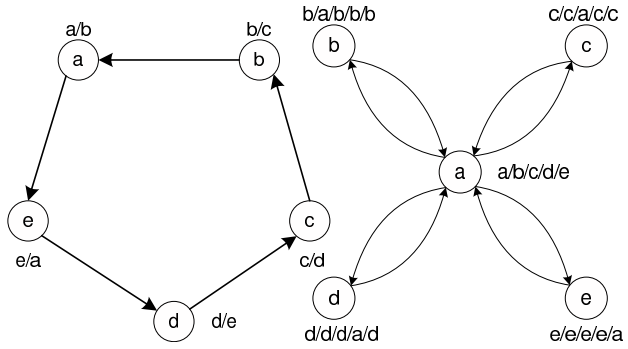
1. Each vertex has exactly one identity assigned to it.
2. Each identity is assigned to exactly one vertex.
3. The identities that can be assigned to a vertex V_i are V_i and the true identities of those vertices that have outbound edges to V_i.

If u is assigned to v in $\mathcal{A}(G)$, we write $u \to v \in \mathcal{A}(G)$.

Similarly, we define individual anonymity.

Definition 3. The individual anonymity $\mathcal{I}(v)$ of a vertex v in an anonymity graph is defined by the number of identities that can be assigned to v.

We use two examples in Figure 3.5 to illustrate the above concepts. In Figure 3.5, possible assignments of identities to each vehicle are marked next to each vehicle. The group anonymity in Example 3.5a is two because there are two possible ways to assign identities: $(a \to a, b \to b, c \to c, d \to d, e \to e)$ and $(b \to a, c \to b, d \to c, e \to d, a \to e)$. Similarly, the group anonymity in Example 3.5b is five. In the following discussion, we use anonymity for group anonymity when the context is specifically about a group. Notice that group anonymity does not necessarily equal the



(a) Example 1 (b) Example 2

Figure 3.5: Anonymities of dynamic groups.

individual anonymity for each member. In Example 3.5b, only a has anonymity 5. The rest have anonymity 2.

Lemma 1. *A vertex v can be assigned exactly those inbound identities that are in the strongly connected component $\mathcal{SCC}(v)$ it belongs to.*

Proof. First, we prove that those inbound identities that are not in the strongly connected component cannot be assigned to v.

Assume that $u \notin \mathcal{SCC}(v)$ and u is assigned to v. We divide the vertices in G into two sets. One set A is the set of vertices that identities have been assigned to. The other set U is the set of vertices that identities are to be determined. We start with putting v into A. Now v has to be assigned to one of the vertices in U according to definition 2.

There are two choices to assign v. If v is assigned to u, v and u form a strongly connected component with two vertices. If v is assigned not to u but to w, we remove w from U and put it into A as illustrated in Figure 3.6.

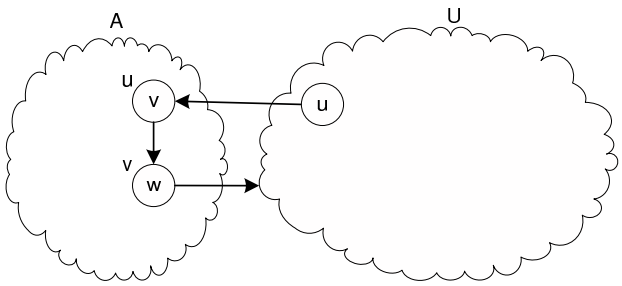


Figure 3.6: Assigning identities.

Now w has to be assigned to some vertex in U. This process continues until we assign an identity to u and u is put into A. Apparently v can reach u following the above path. Thus $u \in \mathcal{SCC}(v)$. This contradicts with our assumption $u \notin \mathcal{SCC}(v)$. Thus those inbound identities that are not in $\mathcal{SCC}(v)$ cannot be assigned to v.

Now we prove that there is at least one unique assignment that assigns each of those inbound identities in $\mathcal{SCC}(v)$ to v.

For any anonymity graph G, there is always an assignment that assigns each identity to its owner. For each inbound identity u of v in $\mathcal{SCC}(v)$, we generate a unique assignment in the following way. At first assign each identity to its owner. Within $\mathcal{SCC}(v)$, there must be a simple cycle containing u and v. Then just shift all the identities along the direction of the cycle to create the new assignment.

When u is different, the assignment is also different since v takes different identities. Thus the generated assignment is unique. □

We have the following Corollaries immediately.

Corollary 1. *The individual anonymity $\mathcal{I}(v) > 1$ if and only if v belongs to a strongly connected component with more than one vertex.*

Proof. This is apparent since $\mathcal{I}(v)$ is exactly the number of inbound identities that can be assigned to v. □

Corollary 2. *The group anonymity has the following properties:*

1. *A vehicle whose identity is not used by other vehicles does not contribute to $\mathcal{K}$. It should be removed from the dynamic group.*
2. *A vehicle who does not use any identity from other vehicles does not contribute to $\mathcal{K}$. It should also be removed from the dynamic group.*

Proof. In both scenarios, the vehicle does not belong to a strongly connected component with more than one vertex. Its individual anonymity is thus 1. It should not be included in the dynamic group. □

Corollary 3. *The group anonymity $\mathcal{K}(G)$ is the product of group anonymities of the strongly connected components of G.*

Proof. From Lemma 1, group anonymities among strongly connected components of G are independent. The number of ways to assign identities is thus the product of the number of ways to assign identities to each strongly connected component of G. □

Since group anonymity is basically the product of group anonymities of strongly connected components, we focus on the group anonymity of a strongly connected graph in the following. In order to facilitate our discussion, we define the *assignment graph* of an anonymity graph as the following:

Definition 4. An assignment graph $\mathcal{M}_{\mathcal{A}}(G) = (V', E')$ of an anonymity graph $G = (V, E)$ represents one assignment $\mathcal{A}$ of identities to vertices, where:

$$V' = V \tag{3.7}$$

$$E' = \{\overrightarrow{uv} : \overrightarrow{uv} \in E, u \to v \in \mathcal{A}\} \tag{3.8}$$

Due to Definition 2, it has the following property: the in-degree and out-degree of a vertex v are either $(0,0)$ or $(1,1)$.

Lemma 2. *In any assignment graph $\mathcal{M}(G)$ of an anonymity graph G, a vertex v is either isolated or belongs to a simple cycle.*

Proof. From Definition 4, if the in-degree and out-degree of v is $(0,0)$, v is isolated. If the in-degree and out-degree of v is $(1,1)$, following the construction procedure illustrated in the proof for Lemma 1, v belongs to a simple cycle. □

Theorem 1. *Let $\mathcal{VCC}(G)$ be the number of vertex-disjoint cycle covers of an anonymity graph G, then:*
$\mathcal{K}(G) = \mathcal{VCC}(G)$

Proof. From Lemma 2, each assignment graph is a vertex-disjoint cycle cover of an anonymity graph G. Thus the number of assignments is the number of vertex-disjoint cycle covers. □

In Figure 3.5b, $\mathcal{I}(a)$ is the same as $\mathcal{K}(G)$. Thus if the identity of a is exposed, the rest of the graph has no anonymity. We define *anonymity criticality* as the criticality of an individual vertex in determining the group anonymity. The formal definition is as follows:

Definition 5. The anonymity criticality $\mathcal{C}(v)$ of vertex v in an anonymity graph G is defined as:

$$\mathcal{C}(v) = \frac{\mathcal{I}(v)}{\mathcal{K}(G)} \tag{3.9}$$

It has the following properties:

1. $0 < \mathcal{C}(v) \leq 1$
2. The greater $\mathcal{C}_v$, the more critical v is in determining $\mathcal{K}_G$.

   For a complete anonymity graph G, $\mathcal{C}_v = 1/(|G|-1)!$.

3.5.2 Application of Mobile Anonymity to Vehicular Networks

We consider the following dynamic group construction scenario in vehicular networks. Then we discuss the application of the general *Mobile Anonymity* model to vehicular networks.

We illustrate the application of *Mobile Anonymity* in Figure 3.7. There are three vehicles a, b, c in Figure 3.7. Vehicles a, b, and c made a request at time T_1 using (a, b), at time T_2 using (b, c), at time T_3 using (b, c), respectively, where $T_1 < T_2 < T_3$. *Spd* is the maximal speed of a vehicle in this scenario.

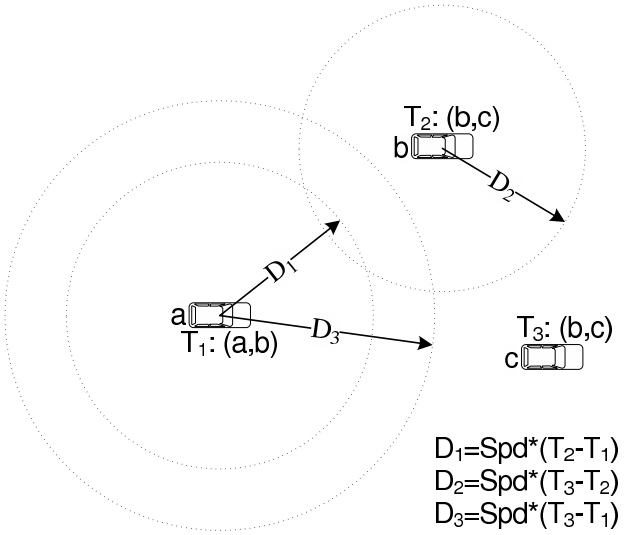


Figure 3.7: Anonymity in vehicular networks.

At time T_2, b is outside the possible reach of a. At time T_3, c is outside the possible reach of both a and b. Thus, a, b, and c can be identified as different vehicles. Its anonymity graph is shown in Figure 3.8. According to Corollary 1, a has no anonymity while the anonymities of both b and c are two.

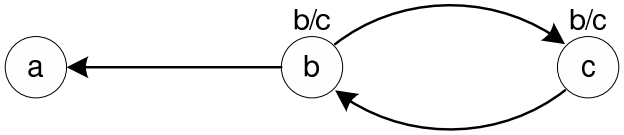


Figure 3.8: Anonymity graph of Figure 3.7.

Figure 3.9 illustrates an example in which two vehicles a and b cannot be told apart. In

Figure 3.9, the three RSUs are separated far enough that the requests made at T_1, T_2, and T_3 can be clearly identified as requests from different vehicles. However, since δ is very small, the request from a at T_3 and the request from b at $T_3 + \delta$ cannot be reliably distinguished as being from two different vehicles.

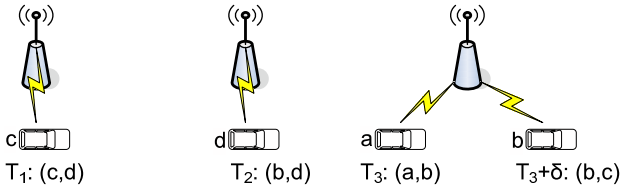
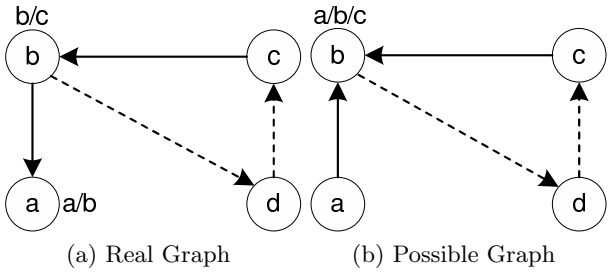


Figure 3.9: Local anonymity in vehicular networks.

Figure 3.10a is its anonymity graph. According to Corollary 1, a should have no anonymity. However, it is also possible that b made both requests. The resulting anonymity graph would be shown in Figure 3.10b. In Figure 3.10b, a is hidden since it did not make any request.



(a) Real Graph (b) Possible Graph

Figure 3.10: Local anonymity graph of Figure 3.9.

In the above examples, we use the assumption that two vehicles using the same certificate can only be reliably told apart when they are over some distance apart within a short period. This is essentially based on the speed limit of a vehicle in a vehicular network. The algorithm is listed in Figure 3.11. There may be other mechanisms to differentiate between two vehicles. However, our approach can be applied similarly with other mechanisms by substituting the function in Figure 3.11.

As a vehicle travels through the road network, the set of certificates collected by it keeps changing over time. The anonymity graph thus changes over time. To deal with this dynamic, we construct a vertex for each request, instead of for each vehicle. We also extend the generalized

Input: two requests $i = (Loc_i, T_i)$ and $j = (Loc_j, T_j)$
Output: $true/false$ whether the two requests can be told apart
$len \leftarrow$ the length of the shortest path between Loc_i and Loc_j
$deltat \leftarrow$ the time difference between T_i and T_j
if *len/deltat is greater than the speed limit* **then**
    return $true$
else
    return $false$
end

Figure 3.11: Function Distinct(i, j).

anonymity graph to represent each request as illustrated in Figure 3.12.

In Figure 3.12a, vehicle IDs d and e are used by vehicles a and c, and b respectively. They are not present in the graph due to the hidden scenario illustrated in Figure 3.9. Figure 3.12b shows its extended anonymity graph, in which we mark each vertex with its associated request information. Figure 3.12c is its reduced form. Since a does not belong to the strongly connected component with b, b will be removed from the set of identities used by a.

We use the algorithm in Figure 3.13 to construct the extended anonymity graph.

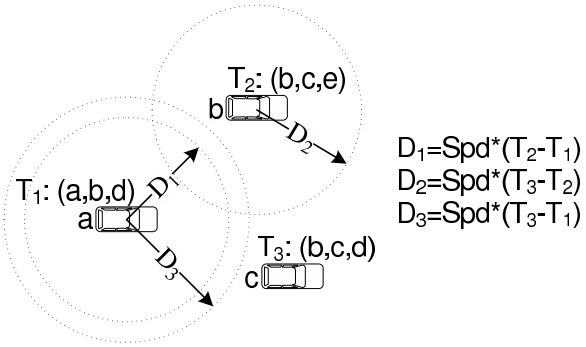
3.5.3 Simulation Evaluation

We run simulations to evaluate the effectiveness of different strategies in preserving anonymity. The simulation is conducted with the C3 simulator [21]. C3 is based on SWANS (Scalable Wireless Ad Hoc Network Simulator) [9].

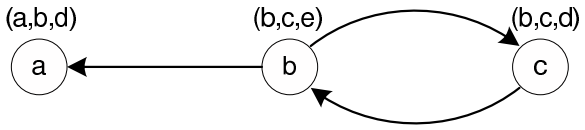
The simulation scenario is set up as follows. We used a lattice road network to simulate a local road network. The size of the network is 7.5 km $\times$ 7.5 km. Each street block is 250m $\times$ 250m. There are totally 100 simulated vehicles. All vehicles are equipped with OBUs. There are totally 16 simulated RSUs that form a 4×4 equidistance grid in the network. We assume that a vehicle makes one request to each RSU. The simulation is run for 20 minutes. The average vehicle speed is 10 meters per second, which is about 22 miles per hour.

We evaluate the following certificate usage strategies by vehicles.

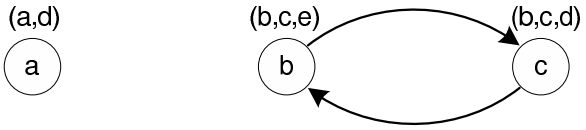
1. **Random strategy (RND)**. In this strategy, each vehicle randomly chooses some fixed



(a) Scenario



(b) Anonymity graph



(c) Reduced Anonymity Graph

Figure 3.12: Extended request-based anonymity graph.

Input: set of requests R made at different RSUs sorted in ascending order of occurrence
Output: the extended anonymity graph
initialize vertex set $V = []$
initialize edge set $E = []$
foreach *Request* $r = (v_r, c_r) \in R$ **do**
    insert r into V
    update the location and time of the vehicle with certificate v_r that initiated r
    `// update out-edges`
    foreach *Request* l *that used* v_r *in the past* **do**
        if $Distinct(l, r)$ *is true* **then**
            insert $\overrightarrow{rl}$ into E
        end
    end
    `// update in-edges`
    foreach *Certificate* $c \in$ *certificate set* c_r **do**
        $l \leftarrow$ last request made by v_c that owns c
        if $Distinct(l, r)$ *is true* **then**
            insert $\overrightarrow{lr}$ into E
        end
    end
end
return $G = (V, E)$

Figure 3.13: Extended anonymity graph construction.

number of certificates to use.

2. **Latest-preferred strategy (LPR)**. In this strategy, each vehicle always chooses certificates that were received most recently.

The purpose of RND is to look at how anonymity can be achieved given that each vehicle may behave differently. In our implementation, we require a vehicle to choose at least one other certificate randomly. A vehicle may choose many certificates up to a specified upper limit. The use of LPR is for comparison under a non-random strategy.

For a vehicle to anonymously distribute its certificate, the broadcasting protocol is similar to the authentication protocol. Instead of just broadcasting its own certificate, a vehicle broadcasts a group of certificates. The broadcasting protocol is more relaxed than the authentication protocol. There is no need for the originator to prove its possession of any private key since certificates are signed by the CA.

Although at the beginning, a vehicle would have to broadcast only its own certificate. However, there is no reliable way to tell whether a vehicle is broadcasting its own certificate or just another certificate. Thus, we do not investigate this case in the following simulation.

Properties of Anonymity Graph

We present the properties of the constructed anonymity graph in the simulation. The specified upper limit for the number of certificates is varied between 2 and 10. For each upper limit, we run the configuration 10 times. For each run, we run Algorithm 3.13 to construct its anonymity graph. For each configuration, we record the sample mean and sample standard deviation of the sizes of the largest components in the 10 runs. The result is shown in Figure 3.14.

The size of the largest component increases as the number of certificates increases. This is because the increased number of certificates creates more edges in the anonymity graph. We do not observe any significant difference between the two strategies. This shows that LPR is not effective in overcoming the randomness in the network. In the subsequent sections, we present only the results obtained with RND. The results obtained with LPR are similar.

Figure 3.15 shows the percentage of non-isolated vertices at different speed limits given to

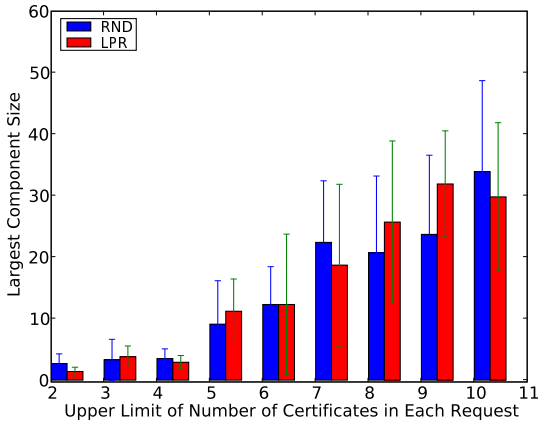


Figure 3.14: Largest component size.

Algorithm 3.11. In the *distinct* case, we intentionally assign -1.0 as the speed limit so that every request can be seen as distinct. In this case, given that 10 certificates are used in each request, the anonymity graph is almost strongly connected. This shows that the use of more certificates will help preserve anonymity even if a random strategy is used. This is a property of random graphs. A random graph is almost connected when its average vertex degree is over a threshold.

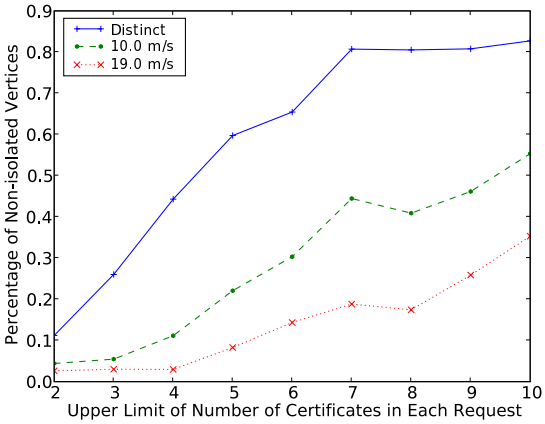


Figure 3.15: Percentage of non-isolated vertices.

Anonymity of Each Request

The reduced anonymity graph in Figure 3.12c shows that a vehicle can still have anonymity even if it does not belong to a strongly connected component. Figure 3.16 shows the average anonymity

for different numbers of certificates. It grows almost linearly with the number of certificates.

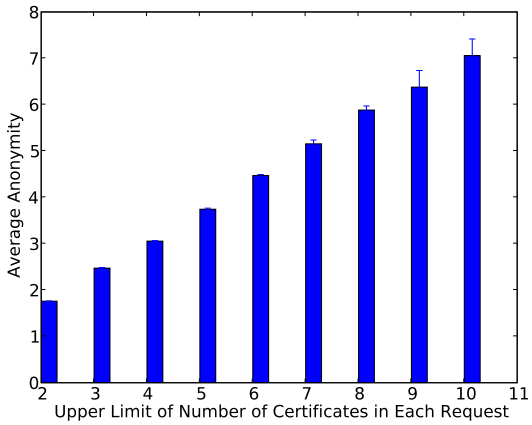


Figure 3.16: Average anonymity.

Figure 3.17 shows the average number of requests that do not have any anonymity. When vehicles use only one other certificate in each request, a significant portion of requests can be identified because the mutual use of each other's certificate between two vehicles is unlikely due to random picking. However, even in this case, only about 24 percent of requests are exposed. Although there are very few requests exposed when the number of certificates is larger, we have verified that no request is exposed if we increase the minimal number of certificates required in each request. We omit its display here because all the numbers are zeros.

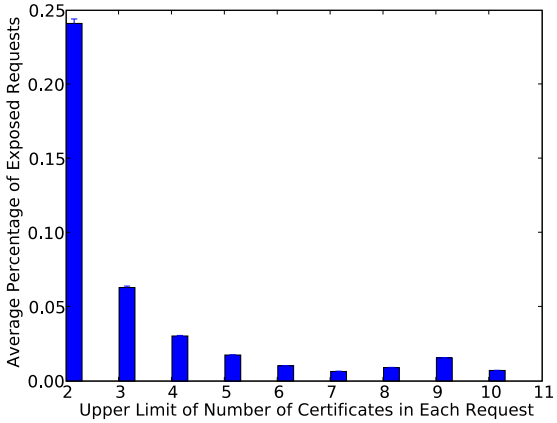


Figure 3.17: Average percentage of exposed requests.

The simulation results clearly support using a large number of certificates in each request. The anonymity of each request in an extended anonymity graph comes from two sources. On the one hand, a larger set of certificates increases the probability of the request belonging to a strongly connected component. On the other hand, the number of requests not distinguishable will also increase with a larger set of certificates.

3.6 Effectiveness of Shared Key Anonymous Authentication Protocol

Symmetric key approaches have been widely used for encrypting/decrypting purposes as they are more efficient than public key based approaches. Computation overhead of symmetric cryptography is much less than that of asymmetric cryptography under current implementations, which is a desired feature in real-time applications, such as vehicular networks.

Symmetric random key-sets are sets of symmetric keys drawn from a shared key pool. It has been applied in wireless sensor networks to establish secure links between sensors because of the high probability of two key sets sharing a common key [29]. We take advantage of the feature that one symmetric key is shared by several vehicles to provide anonymity. We use symmetric key-sets for authenticating vehicles while preserving their privacy. Here, we first introduce the basic idea of symmetric random key-set. Then we provide the detail of the privacy-preserving authentication protocol.

3.6.1 Symmetric Random Key-Set

In the symmetric random key-set approach, all valid keys form a key pool and each member will randomly draw a key set from the key pool. If the size of the key pool and the size of the keys held in each member are properly chosen, there is a high probability that one key is shared by a set of members. In our approach, we take advantage of this feature, i.e., when the keys shared by a set of vehicles are provided to the RSU for authentication purposes, the RSU cannot distinguish the vehicle that requests an authentication, because the same key may be provided by some other

vehicles.

Symmetric random key-set has several advantages in an application such as vehicular networks, which have strong real-time and privacy requirements. First, symmetric keys take much less time in encryption and decryption so that it is more suitable to be applied in a vehicular network which has strong real-time requirement. Second, each key is shared by a set of vehicles; thus, privacy of the vehicle is well protected because the identity and the keys are not closely correlated. Third, each vehicle holding a set of keys is helpful with key revocation, i.e., even if some of the keys have been revoked, the rest of the vehicles can still have a chance to be successfully authenticated. Fourth, the whole key set or a majority of the keys in one vehicle can act as the identity of the vehicle. Thus, if the whole set of keys can be captured by a third party, the identity of the vehicle can be revealed.

The whole set of keys in each OBU is called its key ring. In the rest of the chapter, I use key ring and full identity interchangeably.

3.6.2 Key Pre-Distribution and Revocation

The system architecture of VII is shown in Figure 3.18. It consists of key distribution center(s), service providers, vehicle registration sites, RSUs, and OBUs (i.e., cars). A key distribution center (KDC) serves as the central authority for administrating keys and assisting in identifying a malicious user. RSUs actually authenticate vehicles. OBUs initialize authentications on behalf of users.

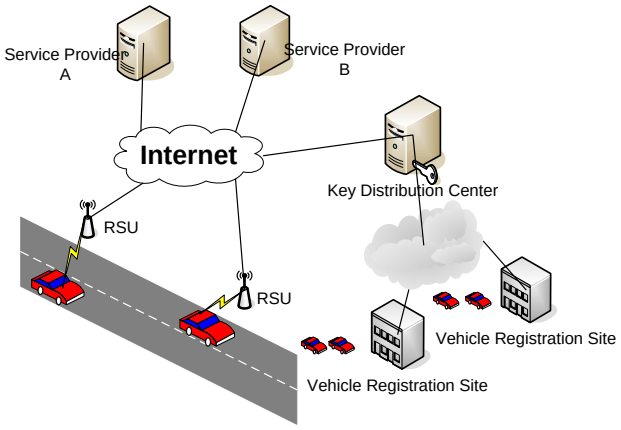


Figure 3.18: A typical system architecture of VII.

In our approach, key pre-distribution is done when the vehicle is sold and when the vehicle is registered with each state. The OBU will be installed with a set of a symmetric keys randomly drawn from the key pool. These keys are used to upload data or download services. They are stored in some well protected components, e.g., tamper-proof devices on vehicles. The keys are renewed every time the registration of the vehicle is renewed in state offices, e.g., DMV in New Jersey. Thus, the symmetric keys have a lifetime of at most one year if the period between registration is one year. The expiration of keys can improve the security of the system. Key pre-distribution simplifies the classical problem of key distribution, which is especially difficult in such a large scale system.

3.6.3 Privacy-Preserving Authentication Protocol

We argue that identity privacy and location privacy are critical to the real deployment of this system when services are downloaded and data are reported. Referring to the privacy level described in [77], we intend to preserve privacy under *zero-trust*, in which the network and servers are not trusted to provide enough privacy protection. In our protocol, we take advantage of a symmetric random key-set, in which every key is shared by a set of vehicles. Thus, when the key is used for authentication, an RSU cannot uniquely identify the OBU because the key may be provided by other vehicles who share the same key.

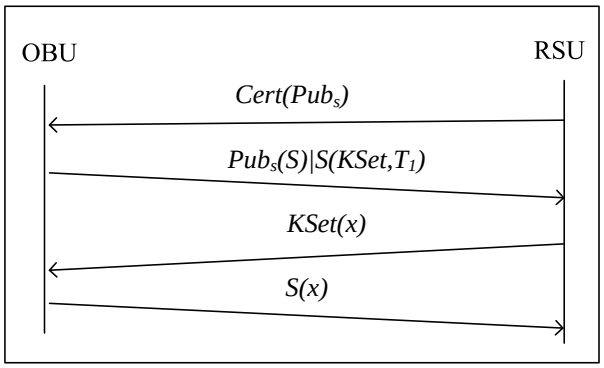


Figure 3.19: The protocol exchange diagram of privacy-preserving authentication protocol using symmetric key-set.

Our privacy-preserving authentication protocol consists of several steps as depicted in Figure 3.19. First, the RSU announces its service by broadcasting certificates signed by the KDC. When

a vehicle decides to access the service, it will send an authentication request together with a set of a key indices that were assigned from the key pool. $KSet$, which is denoted as $K_{k1}, K_{k2}, ..., K_{ka}, ...$, consists of a set of symmetric keys shared by the vehicle and the RSU. A time stamp T_1 is appended for message freshness. All this information is encrypted by a session key S. The session key S is encrypted with the RSU's public key Pub_S. This encryption protects the message integrity and prevents the keys from being disclosed to an outside observer. A set of keys, instead of one key, are used for authentication, because there is a high probability for the OBU to have one key shared by a large number of vehicles, which makes it difficult to identify a malicious vehicle if the key is reported as invalid, while there is only a much lower probability for a set of a keys being shared by a large number of vehicles so that it is much easier to catch a malicious vehicle.

After the RSU gets the authentication request from the vehicle, it creates a challenge message by encrypting a random secret with the set of keys indicated in the request. The encryption should use the Cipher-block Chaining (CBC) mode with multiple encryption keys. The order to use the set of keys is the same as defined in the authentication request. If the roadside server detects some invalid keys or revoked keys, the authentication will fail immediately. Otherwise, the challenge is sent back to the vehicle to verify the actual possession of those keys by the vehicle. It is worth noting that a legitimate user may use some revoked keys for authentication due to the shared key pool. In this case, we show in Section 3.6.4 that the probability of all a keys being shared with a revoked user is very small. The RSU can thus detect that the key ring of the OBU should be updated. The RSU can then broadcast the revocation list to the OBU, which in turn updates its key ring and uses the updated key ring in future authentications without further problems.

Upon receiving the challenge, the vehicle decrypts the challenge with the chosen keys and creates a response by encrypting the random secret with the session key. The response is sent back to the RSU. The RSU verifies the response by comparing the decrypted secret with its original secret. Upon successful verification, the RSU accepts the session key and the vehicle is authenticated.

Each vehicle defines a period in which a single key in its key ring is used only once. Other than that, the key set used for authentication is randomly selected from the key set the vehicle has. Thus, within one period, we can guarantee that the keys used for authentication are different

Parameters	Description
n	The total number of keys in the key pool
v	The total number of vehicles that registered in the system
k	The total number of keys in one vehicle
a	The number of keys that are sent for authentication
C_n^m	The total number of possible different combinations of choosing m different elements out of n different elements
$n!$	Factorial of n
$P\{a+\}$	The probability of two vehicles sharing at least a keys
$P\{a\}$	The probability of two vehicles sharing exactly a keys
$P\{a+\|A\}$	Given a keys, the probability of another vehicle sharing at least those a keys

Table 3.2: Definition of parameters used in the analysis.

every time, which improves the privacy of the protocol as discussed in Section 3.6.4. The session key has only a short lifetime. It is valid in a local area. The policy to manage the session key can be enforced by an OBU according to its privacy requirement. For example, a session key can be geo-bounded. The size of the area may be defined by the application.

3.6.4 Privacy Analysis

In the protocol described above, every time a vehicle enters some area it will send a set of symmetric keys to the roadside server to request an authentication. In this section, we analyze the anonymity provided by our protocol. The parameters we used in this section are listed in Table 3.2.

Static Analysis

Anonymity is the state of being not identifiable within a set of subjects, the anonymity set [67]. To analyze the anonymity provided by our protocol, we have to construct the corresponding anonymity set.

In this subsection, we construct the anonymity set out of the global set. We call this analysis

static analysis since it does not consider any traffic scenario. And we call the anonymity under this scenario static anonymity. Static anonymity is the basic anonymity enjoyed by each vehicle. In a dynamic traffic scenario, the set of vehicles on the road is always a subset of the global set. Thus the anonymity in this dynamic traffic, the dynamic anonymity, is less than the static anonymity because of the smaller number of vehicles. We analyze the dynamic anonymity in the next subsection. The meaning of this static anonymity is that when the network sees that the same set of keys is used for two different authentications, the network can not decide whether it is from the same vehicle or not.

Consider when multiple vehicles are accessing the same RSU. Assume that every vehicle has the same probability to be in the same area. Given a keys, the probability of two vehicles randomly picked out of the whole vehicle set sharing at least those a keys is:

$$P\{a+|A\} \approx \left(e^{-\frac{1}{2n}} e^{\frac{1}{2k}} \left(\frac{k-a}{n-a}\right)\right)^a$$

Assume that the combinations assigned to all vehicles are uniformly picked out of the whole combination space. The real static anonymity for each authentication is $P\{a+|A\}v$.

Dynamic Analysis

We designate the vehicle accessing the network as vehicle Bob. We also want the probability of Bob being tracked to be small. We assume that multiple vehicles are traveling on the same road. This is a reasonable assumption. If there is only Bob on the road, whatever keys he uses, he can still be tracked. Since RSUs do not know exactly what keys are stored in each vehicle, they can only make a combination of all possibilities. We show that anonymity is well preserved under this scenario.

To prevent correlations from being made, the OBU adopts an authentication reuse period concept. Within one period, for each authentication the OBU randomly picks up a keys which have not been used in the same period. So the average period of reusing each key is $c = \frac{k}{a}$ (in number of authentications). Assume that $T(c)$ other vehicles have been accessing the same set of

OBUs during this time. The expected number of vehicles sharing a key with Bob is:

$$T(c) \times (1 - P\{0\}) \tag{3.10}$$

where $P\{0\}$ is the probability of two vehicles sharing no key.

$$P\{0\} \approx e^{-\frac{2k^2}{n}}$$

It can easily be seen from the above equation that the expected number of vehicles increases as k increases. Since $P\{0\}$ is small, the number of vehicles is dominated by $T(c)$. This means that it can easily be increased by extending the period. Figure 3.20 shows the expected number of vehicles for different values of a. We assume that $T(c)$ is linear with respect to $\frac{k}{a}$. This is a conservative assumption. It can easily be seen from this figure that the expected number of such vehicles is quite large.

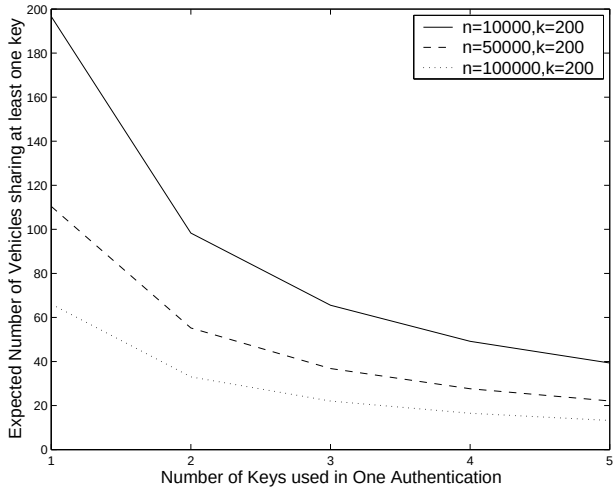


Figure 3.20: The expected number of vehicles sharing at least one key.

Since those vehicles share a key with Bob and we use all keys in a period, it is guaranteed that those authentications have common keys with Bob. This creates diversion if RSUs are trying to correlate the authentications with one common key. We need to point out that the authentications with a common key do not have to happen at the same RSU. Because we use different keys for each authentication within one period for each vehicle, the correlation can not be done for each vehicle

within one period. Essentially, the tracking can only be done based on the whole key ring.

If RSUs are trying to correlate authentications with r common keys, where $1 < r \leq a$, the average time (in number of authentications) of the same r keys being reused for each vehicle can be shown as the following formula:

$$R = \frac{k!r!(a-r)!}{a!a!(k-a)!}$$

R is dominated by $\frac{k!}{a!(k-a)!}$ since r and a are small compared with k. This number is very big given a reasonable value for k. For example, for $k = 200$, $a = 5$ and $r = 3$, its value is $253,565,004$. It is easy to see that it is very unlikely that it will be repeated.

We also point out that it is very difficult for RSUs to correlate between individual authentications since authentications from the same vehicle do not have common keys within one period and authentications from a different vehicle do not have common keys either with high probability. It is very difficult to differentiate between them.

3.6.5 Effect of Key Revocation

Upon a vehicle being reported lost, its key ring may be revoked to prevent possible attacks from this vehicle. To disable possible access from this vehicle, all keys in its key ring will be revoked. This will affect other vehicles since the same key is shared by multiple vehicles. Revoking one key will also prevent legitimate vehicles from using the same key for authentication. However, in the following, we show that the effect is very limited.

The probability of two vehicles sharing exactly a keys is:

$$P\{a\} = \frac{C_k^a C_{n-k}^{k-a}}{C_n^k}$$

And the probability of two vehicles sharing at least a keys is:

$$P\{a+\} = \sum_{i=a}^{k} P\{i\}$$

Since a is small, we compute $P\{a+\}$ as:

$$P\{a+\} = 1 - \sum_{i=0}^{a-1} P\{i\}$$

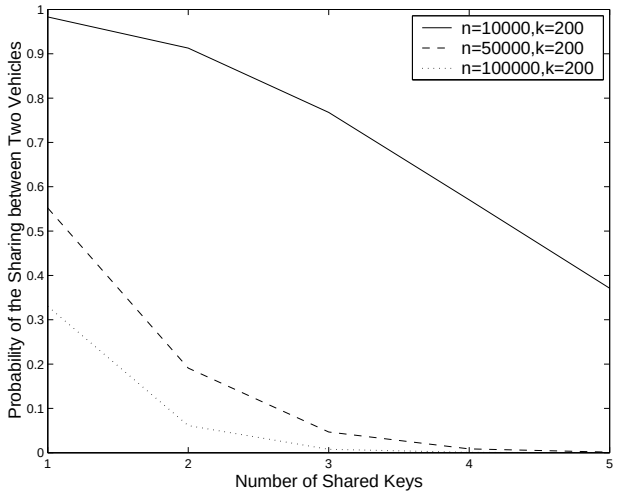


Figure 3.21: The probability of sharing different number of keys between any two vehicles.

The values of $P\{a+\}$ for different values of a are shown in Figure 3.21. It can be seen that for large values of n, the probability of two vehicles sharing more than three keys is very small. Thus revoking one vehicle will have minimum effect on other vehicles. If the KDC detects that another vehicle shares several keys with the vehicle being revoked, the KDC can invite the other vehicle to renew its key ring.

3.6.6 Suggestions on Choosing Parameters

The choosing of parameters is complicated since it involves satisfying multiple constraints. However, we give some suggestions in this subsection. The essential privacy protection of our protocol is provided through a high probability of sharing one key between two random vehicles and a large key ring in each vehicle so that the number of combinations used for authentication is large. Consider that the number of combinations of choosing five keys out of a large key ring is already very large. For practical applications, we suggest that the number of keys used for each authentication is no greater than five. Also, to make the number of combinations large enough, we suggest that it

should be at least three.

The expected number of vehicles sharing common keys with another vehicle is mainly determined by $\frac{k}{a}$. The next step is to choose a proper size k for the key ring in each vehicle. The total key pool size n can always be decided afterward. Delaying the decision for n can even give us one more benefit. The key pool does not have to be fixed at all times. It can grow over time to provide more flexibility. It can also purge expired keys over time. As long as we have a big enough window for all the vehicles, the desired anonymity can still be guaranteed.

Chapter 4

Private Information Retrieval

In the previous two chapters, we have discussed techniques that preserve privacy by either evading detection or removing associated private information. In certain scenarios, it is necessary to preserve the information itself without degrading it, and the process of getting that information cannot be hidden, for example, by direct interaction with an adversary. Preserving privacy appears to be difficult in this scenario. In this chapter, we illustrate such a scenario under the context of online navigation. We show that computational private information retrieval can be used as the model for this scenario and discuss its complexity. Then we discuss some trade-offs to reduce the complexity.

4.1 Introduction

GPS-based navigation is quite common today. It is especially useful when one is visiting a new area. A standalone GPS device contains built-in maps. The map is used for multiple purposes. First, it is used to relate GPS coordinates to the physical context, such as which road the user is currently on. Second, it is used for calculating the shortest path between an origin and a destination. Finally, it often contains additional information such as points of interest so that an user would be able to find places he may be interested in.

In recent years, phone-based navigation is gaining popularity. A typical example is Android-based Google map navigation. There are multiple reasons behind this. First and foremost, cellular phones are becoming much more powerful. GPS is becoming a common component of cellular

phones. The phones also have bigger and better screens. Sometimes their screens are comparable with dedicated GPS screens. Also, they have faster processors and larger memory capacities that are necessary to run a rich application, such as the navigation application. Second, broadband cellular data services are increasingly available and more affordable. This has enabled cheap or free service-based navigation as the map and direction data are retrieved on-demand.

Service-based navigation is a type of location-based service. The origin and the destination information are sent to the server. Based on routing options, such as choosing the fastest or shortest route, a planned route is returned and the navigation application uses the route and location information provided by the GPS to guide the user to the destination.

In location-based services, privacy is a sensitive issue. This certainly applies to service-based navigation as the origin and the destination are most likely associated with the user's personal plans. The privacy is even more sensitive here as it gives a perception of the user's movement being tracked. Krumm [50] gives a thorough review of computational location privacy. Computational location privacy involves interpreting location information as geometric information and quantifying privacy in such a way. For example, it has been shown that by merely looking at a sequence of locations from a single user, a person could be profiled [14].

The common strategies in protecting location privacy are through anonymity and obfuscation. Anonymity-based approaches mix one person's information with another's. It is intended to remove the uniqueness of a person's location information. Obfuscation-based approaches reduce the precision of the location information so that it becomes less sensitive.

Finding the shortest path between an origin and a destination is significantly different from existing location privacy problems. The pair of origin and destination information is necessary input for finding the path. Besides the privacy of each piece of information, the pair itself represents an unique combination that may be personally identifiable. Yet, they are inseparable.

In this chapter, we investigate possible solutions of finding the shortest path between an origin and a destination, without the risk of compromising their privacy. We apply the principle of private information retrieval (PIR) [22] and discuss practical solutions.

The rest of the chapter is organized in the following way. In section 4.2, we present our problem

definition in detail. In section 4.3, we discuss related work. fghgfIn section 4.4, we apply the principle of PIR and discuss their complexity. In section 4.5, we present practical solutions.

4.2 Problem

In this section, we use a transportation network as our case study scenario in studying the privacy-preserving shortest path search. First, we describe the application scenario and its motivation. Then, we abstract the problem, turning it into a privacy-preserving shortest path search.

4.2.1 Application Scenario

Phone-based navigation is gaining popularity as mobile Internet is more affordable and GPS becomes a standard option on smart phones. Typical examples include Google map navigation and Microsoft Bing map navigation. A dedicated GPS navigation device uses a built-in map. In contrast, phone-based navigation software does not have a pre-installed map. It thus depends on the service for actual routing.

From a privacy perspective, using a built-in map has great benefits because it does not disclose origin and destination information. Service-based routing, on the other hand, requires the origin and destination being transmitted in order to find the shortest path between the origin and the destination. Although there could be policy-based approaches to limit access to this information, we are focusing on the problem of not disclosing the origin and the destination in this chapter. We strongly believe that it should be one of the privacy choices that are presented to users.

The lack of a local map on a phone can be partially alleviated by caching map data. This is especially true if the map data is relatively static. By using the cached data, the phone could compute the shortest path locally, thus removing the privacy concern. Similarly, using cached data to protect location privacy has been considered by [56, 64, 20]. It essentially decouples spatial and temporal information and depends closely on the data being relatively static.

In this chapter, we will instead focus on the case when the shortest path is provided by the service. There are two practical reasons for this. First, although smart phones are quite powerful today, finding the shortest path might still involve searching through a large graph, which could

be slow. Second and more importantly, finding the shortest path should inherently be a server-side task. For example, ideal route planning in transportation is highly dependent on real-time traffic information, which suggests that giving path suggestions is a type of mesoscopic scale traffic control. Thus, protecting privacy in this case is an important scenario that we need to tackle.

In this chapter, we will assume that static information, such as the network topology, is already available on the client side. This can either be achieved by caching the topology for later use, or by using preinstalled maps. This is a reasonable assumption as the topology of the transportation network does not change frequently.

4.2.2 The Problem

We formulate the problem as a two party graph searching problem. One party, A, holds the location information of its origin, o, and destination, d. It also holds the topology of the network. However, it does not know the cost of traveling each link in the network. The other party, B, holds the graph in which the origin and the destination are connected and it knows the cost of traveling each link. The problem is for A to find the shortest route from the origin to the destination, without disclosing either the origin or the destination information to B. Formally, A holds the pair (o, d). B holds the graph $G = (V, E)$, where $o \in V$ and $d \in V$. To accommodate our application scenario, we restrict our discussion to simple directed graphs with only positive weights on edges.

Basic Solution

A basic solution is for B to send to A a representation of all-pairs shortest path information upon request. Then A uses o and d to extract the shortest path from the received representation. A does not disclose any information wrt (o, d) in this process.

In this section, we derive the minimum representation of all-pairs shortest path.

Proposition 1. *The minimum representation of all-pairs shortest path in G includes every edge in G.*

This is because shortest paths between neighboring nodes are the edges connecting them in simple directed graphs with only positive weights. Thus all edges need to be included in order to

sufficiently express the shortest paths between neighboring nodes.

Proposition 2. *All-pairs shortest path can be sufficiently expressed by a path reconstruction matrix as illustrated in the Floyd-Warshall algorithm.*

This is straightforward from the Floyd-Warshall algorithm. The shortest path is recursively constructed by looking up values in the matrix with the current node and the destination and finding the next hop.

4.3 Related Work

A number of works [20, 64, 56, 6] apply caching to location-based services. The basic rationales for using caching to protect privacy include: first, it decouples the physical location and information needed for that location in time; second, the estimation during caching often encompasses a larger area, which serves to camouflage the precise physical location. Our approach complements those approaches and explores the scenario where caching cannot be applied, such as when precise real time information is needed.

There are two types of PIR techniques [22, 94]. Information theoretic private information retrieval gives unconditional protection for the privacy. Approaches with better communication cost than the trivial solution of sending over the entire information set usually use multiple non-colluding parties holding copies of the data set. On the other hand, with only a single source of data, asymptotically better solutions have been obtained through the assumption of computationally hard problems. However, Sion and Carbunar [80] argued that existing cPIR approaches are not computationally practical compared to the trivial solution. Thus they have proposed a solution based on Oblivious RAM [92]. In this chapter, we analyzed the complexity of applying cPIR to finding the shortest path between a origin and a destination. We also proposed a practical solution that uses a third party, which acts similarly as Oblivious RAM. Thus our work complements existing work.

There are many other works that provide practical solutions on privacy-related issues. Song et al. [82] proposed a solution to search for a word privately on the server. The application scenario

is for a user storing encrypted emails on a server. In this application, the user would generate the encrypted emails in the first place. This is different from our scenario in which the service provider is the information source. Bellovin and Cheswick [11] provided a solution using Pohlig-Hellman encryption for querying an encrypted Bloom filter. A Bloom filter is good to query for membership but does not provide additional information. In our application scenario, the next hop needs to be retrieved and at the same time we prevent the third party from applying graph analysis by obfuscating the graph.

There is also a class of problems that deal with data from third parties that is encrypted with a user's public key and is only intended for a user to consume. Boneh et al. [15] use Identity-based encryption so that it is possible for the mail server to test whether an incoming encrypted email contains a keyword or not. Our problem still belongs to the public data source domain. Thus PIR is a more appropriate technique to apply.

Recently, Olumofin and Goldberg [62] show that multi-server information-theoretic PIR is not subject to the computational constraints of cPIR and a recently proposed single-server lattice-based PIR outperforms existing cPIR significantly. As our problem domain is in PIR, we will benefit from any further advance in this field.

4.4 Private Graph Search

In this section, we apply single database, computationally-private information retrieval (cPIR) to our graph search problem. We chose the single database cPIR algorithm because in practice it is hard to satisfy database replication across different vendors. First, we briefly review the algorithm. Then, we describe its application to our graph search problem.

4.4.1 Computationally-Private Information Retrieval

In computationally-private information retrieval, a user wants to retrieve a piece of information from a single public database, without revealing which piece he retrieved. For discussion purposes, a database is assumed to be an n-bit string and the user wants to retrieve the i-th bit. Computationally-private information retrieval guarantees that the database performs only

polynomial-time computation and the identity of i is hidden from the database.

Eyal and Rafail proposed the first single database, computationally-private information retrieval algorithm (ER02) [51]. The algorithm relies on the hardness of deciding quadratic residuosity. It works in the following way:

1. The user randomly picks two large, $k/2$-bit primes, where k is the security parameter. He calculates the result of multiplying those two numbers - a k-bit number N. He keeps the factors secret.

2. Based on the number of bits n, the user picks a recursion depth $L = O(\sqrt{\log n / \log k})$.

3. The user randomly picks $t = n^{1/(L+1)}$ numbers $y_{11}, \ldots, y_{1t} \in Z_N^{+1}$, where $Z_N^{+1} = \left\{ y \in Z_N^* | \left(\frac{y}{N}\right) = 1 \right\}$. Here $Z_N^* = \{x | 1 \leq x \leq N, gcd(N, x) = 1\}$ and $\left(\frac{y}{N}\right)$ is the Jacobi symbol. Of those numbers, y_{1j_1} is a quadratic residue modulo N and all others are quadratic non-residues modulo N. We write $Q_N(y_{1j_1}) = 0$ and $Q_N(y_{1m}) = 1$ if $m \neq j_1$. Here $j_1 = i \mod t$.

4. For each recursion level l where $2 \leq l \leq L$, the user randomly picks another t numbers y_{l1}, $\ldots$, y_{lt}. This time, the user computes $j_l = (i/t^{l-1}) \mod t$ and selects $Q_N(y_{lj_l}) = 0$ and $Q_N(y_{lm}) = 1$ if $m \neq j_l$.

5. The user sends $(N, t, L, y_{11}, \ldots, y_{1t}, \ldots, y_{L1}, \ldots, y_{Lt})$ to the database.

6. The database organizes n bits into a $t^L \times t$ matrix M. It adds padding 0s when necessary. It computes for every r a number $z_r \in Z_N^*$ as follows. It computes this first (for $1 \leq j \leq t$):

$$w_{r,j} = \begin{cases} y_{1j}^2 & \text{if } M_{r,j} = 0 \\ y_{1j} & \text{if } M_{r,j} = 1 \end{cases}$$

It then computes:

$$z_{1r} = \prod_{j=1}^{t} w_{r,j}.$$

This results in $s = t^L$ k-bits numbers z_{11}, $\ldots$, z_{1s}.

7. The database decomposes $z_{11}, \ldots, z_{1s}$ into k t^L-bits strings by grouping the d-th significant bit from $z_{11}, \ldots, z_{1s}$ into x_{1d}. It again organizes x_{1d} into a $t^{L-1} \times t$ matrix. It again computes t^{L-1} k-bit numbers in the same way as above, but with $y_{21}, \ldots, y_{2t}$. It performs this computation for all k strings. This will result in $t^{L-1} \times k$ k-bit numbers, i.e., k^2 t^{L-1}-bit strings.

8. The database repeats the previous step for all recursion levels. The final results are k^L t-bit strings. It sends those strings to the user.

9. The user divides k^L $n^{1/(L+1)}$-bit strings into groups with k strings. It recovers k^{L-1} groups of $n^{1/(L+1)}$ k-bit numbers, by taking one bit from the $n^{1/(L+1)}$-bit string within each group. It then chooses the i/t^L-th number from each group. Since the user knows the factors of N, he is able to efficiently determine whether each number is a quadratic residue or not by calculating the Jacobi symbol of the number to each prime factor. If the number is a quadratic residue, he substitutes the number with bit 0, otherwise with bit 1. This results in k^{L-1} bits. The user groups them into k^{L-2} k-bit numbers. Again, he is able to efficiently reduce each number to bit 0 or bit 1. The user keeps doing this recursively, until he obtains 0 or 1 at the end, which is what he needed from the database.

The communication cost between the user and the database is:

$$\begin{aligned} D &= t\left(k^L + L \cdot k\right) \\ &= O(tk^L) && as \quad L \geq 1 \\ &= O(n^{\frac{1}{L+1}} k^L) && as \quad t = n^{1/(L+1)} \\ &= O(k^{L+\frac{L^2}{L+1}}) && as \quad n = k^{L^2} \\ &= O(k^{2L}) \end{aligned}$$

The computation cost on the client side is mainly the cost of recovering the requested bit from the response. By measuring the cost in the number of Quadratic Residue verifications, that cost is

$O\left(k^L\right)$. The computation cost on the database side, in the number of multiplications, is:

$$\begin{aligned} C &= O\left(\frac{t\left(t^{L+1}-k^{L+1}\right)}{t-k}\right) \\ &= O\left(\frac{k^L\left(k^{L(L+1)}-k^{L+1}\right)}{k^L-k}\right) \quad as \quad t = k^{\frac{L^2}{L+1}} \\ &= O\left(k^{L^2+L}\right) \end{aligned}$$

4.4.2 Decision Table

In this section, we model the shortest path problem as a one dimensional database. The approach we take is to build a four-column decision table, in which the first column is the current position where the client is, the second column is the destination, the third column is the intermediate node, and the last column is a one-bit decision whether the intermediate node is on the shortest path between the origin and the destination or not.

The client finds the shortest path to the destination by always querying for the next hop it should take. Since the client knows the candidate set for the next hop, it queries the server for those entries.

The size of the table (the number of rows) is n^3. For each query, this leads to a communication complexity of $O\left(k^{2\sqrt{3}L}\right)$. Computation cost on the client side is $O\left(k^{\sqrt{3}L}\right)$. Computation cost on the server side is $O\left(k^{3L^2+\sqrt{3}L}\right)$. The total cost of finding the whole path is a linear multiplication of the size of candidate set on the whole path.

4.4.3 Modified Decision Table

In the modified decision table, instead of having a four-column decision table, we have a three-column decision table. The first two columns are the same. The last column is a binary representation of the next hop, an $O(\log n)$-bit number as we have n nodes in the network.

For the client to find the shortest path to the destination, it queries the server directly using its current position and the destination. Instead of the candidate set, it now needs to run the query $O(\log n)$ times in order to get the next hop. However, as the size of the table is n^2, for each query, the communication complexity is $O\left(k^{2\sqrt{2}L}\right)$. The computation cost on the client side is $O\left(k^{\sqrt{2}L}\right)$.

The computation cost on the server side is $O\left(k^{2L^2+\sqrt{2}L}\right)$.

The costs associated with the modified decision table are less than those in the original decision table. For example, the ratio between the communication cost of the original decision table and that of the modified decision table is:

$$\begin{aligned} \frac{k^{2(\sqrt{3}-\sqrt{2})L}}{\log n} &\approx \frac{k^{0.64L}}{\log n} \\ &\gg 1 \qquad \text{as} \quad L \geq 1, n \ll 2^k \end{aligned}$$

$n \ll 2^k$ because in order for k to be a good security parameter, the numeric space a k-bit binary number could represent is much larger than n. For example, k may be 512 to be considered sufficient, while $\log n$ would be 20 for a network with a million nodes.

4.4.4 Path Reconstruction Matrix

In this approach, the all-pairs shortest path is expressed by a path reconstruction matrix. Assuming that there are n nodes in the graph, nodes are labeled with unique $log(n)$ bit numbers $1 \ldots n$. The path reconstruction matrix X is an $n \times n$ matrix. Rows of the matrix are the origins and columns of the matrix are the destinations. x_{ij} is the next hop on the shortest path from i to j.

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nn} \end{pmatrix}$$

The algorithm works in the following way:

1. The user randomly picks n numbers $x_1, \ldots, x_n$ where $Q_N(x_j) = 0$ and $Q_N(x_m) = 1$ if $m \neq j$. He sends those numbers to the server.

2. The matrix is split into $\log n$ bit matrices B_1, B_2, ..., B_k. Each bit matrix B_l stores the lth bit of X.

3. Next, we apply computationally-private information retrieval to each bit matrix B_l. Instead

of retrieving a specific bit, we always retrieve the jth column of each bit matrix, where j is the destination. This is done by setting the recursion level to 1 in the algorithm described in section 4.4.1.

4. The user is able to recover the whole j-th column of each bit matrix. He then reconstructs the j-th column of the path reconstruction matrix. The shortest path from i to j starts from the i-th row of the column, which is the index of the next hop after i. Then, that number is used as the index into the column to retrieve the next hop. The process continues until it reaches j.

The communication complexity of the algorithm is $O(nk \log n)$. The computation complexity on the server side is $O(n^2 \log n)$. The computation cost on the client side is $s \log n$ where s is the length of the shortest path between i and j. The costs are expensive when n is large.

4.5 Private Matching and Graph Obfuscation

Under current PIR algorithms, the cost to privately retrieve a route is too high to be used practically. This is because the mobile communication link between the user and the database is often limited. The long delay incurred with transmitting the information would render the system unusable.

We note that as the PIR technique progresses, the costs may become acceptable in the future. In this section, in recognizing the hardness of current PIR, we propose to use the most basic form of PIR for practical applications: transmitting the entire database. However, to avoid the same high transmission cost we mentioned before, we propose to keep the transmission within the cloud itself. This is feasible as the bandwidth in the Internet backbone is much higher than that on a wireless edge connection. The latter is subject to more physical constraints. For example, for a graph with 10000 nodes, assuming 8 bytes are used to represent one node, the matrix is about 800 MB, which would take less than 10 seconds to transmit on a Gigabit Ethernet link, but would take over 10 minutes to transmit on a broadband 1 Mb mobile link, if not considering potential packet retransmissions that often exist on mobile links.

At the same time, to provide privacy, we propose to separate the generation of the shortest

path and matching of a shortest path with a origin/destination pair. In this scheme, the service provider runs a computing engine and is responsible for calculating the shortest paths. It then sends an obfuscated version of that to a third party that runs an extracting engine who is specifically responsible for performing the private matching on behalf of the user.

The first technique is to use private matching. With this technique, the third party retrieves the shortest path needed by the user from the obfuscated graph without knowing the actual origin and destination. The second technique is an obfuscated graph. The graph generated by the computing engine cannot be used by the third party to discover the origin and destination. In summary, we separate the problem of finding the shortest path for a specific origin and destination pair into two parts: computing the shortest paths and matching the origin and destination against a shortest path. We note that the cost associated with private retrieval can be offset by charging the user, comparing to the free non-private retrieval of the route. We will see from next section that this is doable as authentication is performed during this process.

To support private matching and efficient retrieval of the shortest path, the data structure is the next hop matrix we showed before. To support graph obfuscation, the matrix is extended by adding an additional row and column, which store the indices of the row and column. By using the next hop matrix, the shortest path can be efficiently recovered by tracing through the next hop chain.

4.5.1 Security Model

The security model is illustrated in the following graph:

The mobile terminal sends security parameters to the computing engine, which computes the shortest paths for all pairs of origin and destination. The graph, which is represented as the next hop matrix illustrated before, is obfuscated and sent to the extracting engine, who is responsible for the private matching between the user request and the obfuscated graph. The computing engine returns a ticket to the mobile terminal. The ticket is the ID of the obfuscated graph so the extracting engine can match a user request with a graph.

The security model assumes that there is no collusion between the computing engine and the

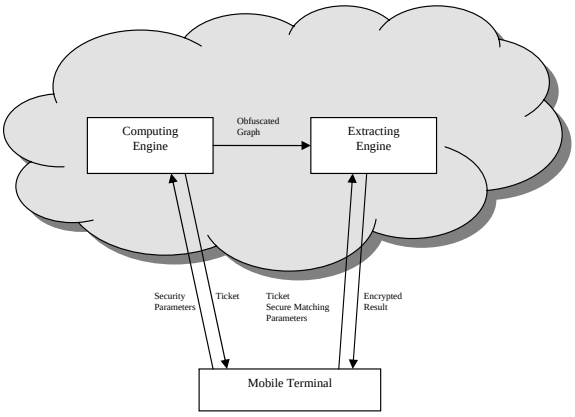


Figure 4.1: Security Model.

extracting engine. This is a reasonable assumption as it is a delegation pattern, in which the matching is delegated to the extracting engine. Delegation is a commonly used security primitive. For example, Kerberos authentication issues delegation tickets so that a user can gain access to different systems with only a single sign-on.

In this security model, the user never discloses the origin and destination information. Yet, at the same time, the high transmission cost is avoided and only information relevant to the actual route is securely transmitted back to the mobile terminal. The only requirement is that there is no collusion between the computing engine and the extracting engine. The information transimitted between the mobile terminal and both parties and that between the computing engine and extracting engine are subject to intercept and analysis.

4.5.2 Protocol

In this section, we first list the security primitives that are required for our scheme. The nodes are represented with n-bit strings $M_{ij} \in N : (0|1)^n$.

1. An identity held by the computing engine and the associated public/private keys: Pub_k and Pri_k.

2. Asymmetric encryption function: $F : Dec_F(Pri_k, Enc_F(Pub_k, Z)) = Z$.

3. A symmetric encryption function $E : K_E \times Z \rightarrow Z$.

The protocol works in the following way:

1. Mobile terminal generates an symmetric encryption key K.

2. Mobile terminal verifies the identity of the computing engine and sends $Enc_F(Pub_k, K)$ to the computing engine.

3. The computing engine encrypts the matrix with K by applying E over every element in the matrix: $M'_{ij} = K \times M_{ij}|M_{1j|M_{i1}}$. The concatenation is to guarantee that a node would have different values at different row/column locations.

4. The computing engine randomly shuffles the matrix M' and sends it to the extracting engine. The extracting engine returns an ID.

5. The computing engine returns the ID to the mobile terminal.

6. The mobile terminial calculates an encrypted query key for a value M'_{ij} in the following way: $Q(i,j) = (K \times i|i, K \times i|j)$. It first calculates this for (o,d) and sends $Q(o,d)$ to the extracting engine.

7. The extracting engine finds the row whose first element matches the first element $Q(o,d)$. It then finds the column whose first element matchs the second element $Q(o,d)$. It returns the corresponding value at the specified row and column to the mobile terminal.

8. The mobile terminal calculates the actual next hop t by decrypting the value with K. It then repeats the above step by calculating $Q(t,d)$, until $t = d$.

4.5.3 Security Analysis

Since the query key is encrypted with K that is unknown to the extracting engine, it is not possible for the extracting engine to know the actual origin and destination. Instead, it does opaque matching of the binary values for matching against the correct row and column.

The graph is obfuscated with two methods. First, the actual matrix is encrypted with the user-provided key K. Since each key is different, the extracting engine cannot perform a dictionary

attack as K is unknown to him. Second, each value in the matrix is concatenated with its unique location, and then encrypted with K. Thus it is not possible to correlate with a different value, unless K is known. This graph is called an obfuscated graph. From the obfuscated graph, the third party matcher cannot recover the topology of the original graph. In fact, the matrix would just look like disjoint points.

Note that although it appears that computation would be expensive at the computing engine, optimizations are possible to minimize its impact. For example, the computing engine keeps a plain copy of the matrix and is ready to generate the obfuscated graph given K. There is no need for it to rerun the shortest path algorithms as the all-pairs shortest path graph is shareable.

Using asymmetric cryptography and verifying identity of the computing engine are necessary. This is because the computing engine is trusted with K, which is crucial for protecting the privacy of the origin and destination. Without identity verification and support of asymmetric cryptography, the extracting engine can perform a man-in-the-middle attack. Privacy would be compromised as it possesses both K and $Q(o, d)$ and can decrypt $Q(o, d)$ to recover (o, d). With asymmetric encryption, only the computing engine can obtain K with its private key.

Chapter 5

Conclusion

Privacy preserving in diversified scenarios is difficult to define and defend at the same time. It is often necessary to make specialized solutions for defined scenarios. Yet at the same time, there are common patterns that can be applied to classes of problems. Wireless sensor network represents a typical example of a class of systems where the system is deeply embedded within the physical space. Thus certain functionality of the network is strongly related with the physical properties of the network. For example, in wireless sensor networks, routing is strongly related with geographical distribution of the routing nodes. Authentication is another class of problems. In mobile networks, authentication is generally required to establish the identities of interacting parties. However, similar to wireless sensor networks, mobile network (especially short range radio-based network) puts the user into a geographical context. This means that the identity established by authentication would tag a location with personally identifiable information. For example, vehicular network relies on Dedicated Short Range Communications (DSRC) for communicating with Road Side Units (RSUs). The last class of problems are those in which private information is disclosed in order to obtain services. This can broadly apply to online services. The fundamental issue with it is whether the private information is necessary for obtaining the service.

For the above classes of problem, it turns out that we can also apply a set of basic techniques that are based on the following principals: evading traffic analysis, anonymity, and private information retrieval. To facilitate discussion, our discussion has been centering around typical emerging net-

work applications. Depending on the context, the investigation into those applications has shown that privacy preserving is necessary in different aspects of a system. In each scenario, we have shown innovative approaches. Below, we summarize our findings, approaches, and future work.

Wireless sensor networks are susceptible to traffic analysis. It has been shown that an adversary can backtrack where packets are coming from to find out the location of a source. Existing approaches suffer from higher energy consumption and statistical analysis attacks. We proposed a bi-directional random walk approach between the source and the destination. Being random at the destination, this means that there is no information about the source being available to the adversary. To improve the efficiency of the random walk, a space-efficient data structure based on Bloom filter is used to avoid local looping. The proposed approach mitigates statistical analysis attack as it limits the amount of communication and reduces energy consumption.

Treating the routing as a function of the source and destination, the protocol is essentially a randomized function. Thus it becomes possible to analyze the function and evaluate the tradeoff between privacy and energy consumption. We leave that to be a future work. Also, although the approach is set in the context of wireless sensor networks, it can be reasonably extended to general multi-hop wireless ad-hoc networks.

A vehicular network is a type of mobile network, in which vehicles are nodes of the network and contribute information to the network. The privacy concerns are from the nodes themselves as each one of them represents a private person. The driver of the vehicle does not wish to be personally identifiable when the vehicle is participating in the network. Yet, the system needs some accountability, especially when certain applications in vehicular network are critical applications that rely on trusted information. Thus in this application, authentication is mandatory. However, from privacy point of view, the downside is that each vehicle now becomes identifiable within the context.

Building on top of asymmetric cryptography, we proposed a k-anonymity authentication scheme, in which a vehicle is identifiable only as a group of k vehicles. This provides limited accountability so additional measures may be taken for legitimate reasons. Yet at the same time, it protects the exact identity. The parameter k is also tunable so that it could vary depending on an actual

application.

The anonymity protocol is extended to mobile sceanrios. Even the authentication is anonymized, it is still possible to track a vehicle by following an unique authentication signature. This is undesired private information disclosure. We term guaranteed anonymity under this scenario as mobile anonymity. We show that mobile anonymity can be quantified by a graph-based approach and derive the necessary conditions for mobile anonymity.

We have also shown that the anonymized authentication protocol can be similarly implemented with symmetric cryptography. This provides a good alternative in scenarios where lower cost in performing cryptography is desired.

The basic strategy we have adopted is to prevent leaking information in undesired ways. However, to fully address privacy concerns, especially when information collection is increasingly pervasive and centralized, it is necessary to look at the possibility of protecting private information itself. For example, the emerging cloud-based systems often process the exact private information sent to them. We illustrate online-based navigation as such an example in which the private origin and destination are shared. We model the problem as a problem of finding the shortest path between a origin and a destination without disclosing the origin and destination to protect privacy. We show that private information retrieval can be applied to it. We deduce the computational complexity of several algorithms based on computational private information retrieval. Recognizing the practical limitation of current cPIR algorithms, we have proposed a practical solution that handles the generation of shortest paths and the match of a shortest path with the origin and destination, respectively, at two different parties. To the best of our knowledge, this is the first work that tries to solve the problem of finding the shortest path without disclosing any information about the origin and destination. It is possible to apply multi-server information-theoretic PIR techniques to the problem so we no longer need to rely on a semi-honest third party. We leave that to our future work.

APPENDIX A: LIST OF PUBLICATIONS

Yong Xi, Weisong Shi, and Loren Schwiebert, Preserving Source Location Privacy in Monitoring-Based Wireless Sensor Networks, International Workshop on Security in Systems and Networks, 2005.

Kewei Sha, Yong Xi, Weisong Shi, and Loren Schwiebert, Adaptive Privacy-Preserving Authentication in Vehicular Networks, International Workshop on Vehicle Communication and Appliations, 2006.

Yong Xi, Kewei Sha, Weisong Shi, and Loren Schwiebert, Enforcing Privacy Using Symmetric Random Key-Set in Vehicular Networks, The International Symposium on Autonomous Decentralized System, 2007.

Yong Xi, Kewei Sha, Weisong Shi, and Loren Schwiebert, Probabilistic Adaptive Anonymous Authentication in Vehicular Networks, Journal of Computer Science and Technology, Issue 6, Vol. 23, 2008.

Yong Xi, Weisong Shi, and Loren Schwiebert, Mobile Anonymity of Dynamic Groups in Vehicular Networks, Wiley's Security and Communication Networks Journal, Volume 1, Issue 3, 2008.

REFERENCES

[1] Auto theft key statistics. `http://www.iii.org/`.

[2] IEEE 1609.2: Trial-use standard for wireless access in vehicular environments-security services for applications and management messages, 2006.

[3] Alessandro Acquisti, Stefanos Gritzalis, Costos Lambrinoudakis, and Sabrina di Vimercati, editors. *Digital Privacy: Theory, Technologies, and Practices*. Auerbach Publications, 2007.

[4] Anne Adams. Multimedia information changes the whole privacy ball game. *Computers, Freedom, and Privacy*, 2000.

[5] Rakesh Agrawal and Ramakrishnan Srikant. Privacy-preserving data mining. In *Proceedings of the ACM SIGMOD ICMD*, 2000.

[6] Shahriyar Amini, Janne Lindqvist, Jason Hong, Jialiu Lin, Eran Toch, and Norman Sadeh. Caché: Caching location-enhanced content to improve user privacy. In *Proceedings of the 9th international conference on Mobile systems applications and services MobiSys 11*, 2011.

[7] Annie I. Antón, Elisa Bertino, Ninghui Li, and Ting Yu. A roadmap for comprehensive online privacy policy management. *Communication of the ACM*, pages 109 – 116, July 2007.

[8] Gildas Avoine and Philippe Oechslin. RFID traceability: A multilayer problem. In *Financial Cryptography (FC'05)*, 2005.

[9] Rimon Barr. *An efficient, unifying approach to simulation using virtual machines*. PhD thesis, Cornell University, 2004.

[10] Victoria Bellotti and Abigail Sellen. Design for privacy in ubiquitous computing environments. In *the Third European Conference on Computer Supported Cooperative Work (ECSCW'93)*, 1993.

[11] Steven M. Bellovin and William R. Cheswick. Privacy-enhanced searches using encrypted bloom filters, 2004.

[12] Alastair R. Beresford and Frank Stajano. Location privacy in pervasive computing. *IEEE Pervasive Computing*, 2, 2003.

[13] Oliver Berthold, Hannes Federrath, and Stefan Köpsell. Web mixes: A system for anonymous and unobservable internet access. *Lecture Notes in Computer Science*, 2009:115–129, 2001.

[14] C. Bettini, X. SeanWang, and S. Jajodia. Protecting privacy against location-based personal identification. In *Proceedings of 2nd VLDB Workshop on Secure Data Management (SDM)*, volume 3674/2005 of *Lecture Notes in Computer Science*, pages 185–199. springer, 2005.

[15] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 506–522. Springer, 2004.

[16] Andrei Broder and Michael Mitzenmacher. Network applications of bloom filters: A survey. In *Allerton Conference*, 2002.

[17] G. Calandriello, P. Papadimitratos, A. Lloy, and J.-P. Hubaux. Efficient and robust pseudonymous authentication in VANET. In *Proceedings of the 4th ACM international workshop on Vehicular ad hoc networks*, 2007.

[18] Murat Caliskan, Daniel Graupner, and Martin Mauve. Decentralized discovery of free parking places. In *Proceedings of the 3rd ACM international workshop on Vehicular ad hoc networks*, pages 30–39, 2006.

[19] Shuchi Chawla, Cynthia Dwork, Frank McSherry, Adam Smith, and Hoeteck Wee. Toward privacy in public databases. In *TCC05*, 2005.

[20] Yu Chen, Jie Bao, Wei-Shinn Ku, and Jiun-Long Huang. Cache management techniques for privacy preserving location-based services. In *Proceedings of the 2008 Ninth International Conference on Mobile Data Management Workshops*, pages 88–96, Washington, DC, USA, 2008. IEEE Computer Society.

[21] David R. Choffnes and Fabin E. Bustamante. An integrated mobility and traffic model for vehicular wireless networks. In *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*, September 2005.

[22] Benny Chor, Eyal Kushilevitz, Oded Goldreich, and Madhu Sudan. Private information retrieval. *J. ACM*, 45:965–981, November 1998.

[23] Crypto++ library 5.2.1, 2006. `http://www.eskimo.com/~weidai/cryptlib.html`.

[24] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type III anonymous remailer protocol. In *IEEE Symposium on Security and Privacy*, 2003.

[25] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *the 13th USENIX Security Symposium*, 2004.

[26] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *the twenty-second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, 2003.

[27] Atakan Doğan, Gökhan Korkmaz, Yiting Liu, Füsun Özgüner, Ümit Özgüner, Keith Redmill, Oscar Takeshita, and K. Tokuda. Evaluation of intersection collision warning system using an inter-vehicle communication simulator. In *The 7th International IEEE Conference on Intelligent Transportation Systems*, 2004.

[28] Tamer ElBatt, Siddhartha K Goel, Gavin Holland, Hariharan Krishnan, and Jayendra Parikh. Cooperative collision warning using dedicated short range wireless communications. In *Proceedings of the 3rd ACM international workshop on Vehicular ad hoc networks*, 2006.

[29] L. Eschenauer and V. D. Gligor. A key management scheme for distributed sensor networks. In *Proceedings of ACM CCS'02*, October 2002.

[30] Keri A. Funderburg. Update on intelligent vehicles and intersections. *Public Roads*, 67, 2004.

[31] Chandana Gamage, Ben Gras, Bruno Crispo, and Andrew S Tanenbaum. An identity-based ring signature scheme with enhanced privacy. In *Proceedings of Securecomm and Workshops*, pages 1–5, 2006.

[32] Buğra Gedik and Ling Liu. Location privacy in mobile systems: A personalized anonymization model. In *Proc. of the 25th International Conference on Distributed Computing Systems*, June 2005.

[33] Philippe Golle, Dan Greene, and Jessica Staddon. Detecting and correcting malicious data in VANETs. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 29–37, 2004.

[34] Victor Gradinescu, Cristian Gorgorin, Raluca Diaconescu, Valentin Cristea, and Liviu Iftode. Adaptive traffic lights using car-to-car communication. In *Vehicular Technology Conference (Spring)*, 2007.

[35] Jonathan Grudin and Eric Horvitz. Presenting choices in context: Approaches to information sharing. In *Workshop on Ubicomp communities: Privacy as Boundary Negotiation*, 2003.

[36] Marco Gruteser and Dirk Grunwald. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proceedings of the First International Conference on Mobile Systems, Applications, and Services (MobiSys)*. USENIX, 2003.

[37] Marco Gruteser and Dirk Grunwald. Enhancing location privacy in wireless LAN through disposable interface identifiers: A quantitative analysis. *Mobile Networks and Applications*, 10, 2005.

[38] Jinhua Guo, John P. Baugh, and Shengquan Wang. A group signature based secure and privacy-preserving vehicular communication framework. In *Proceedings of the Mobile Net-*

working for Vehicular Environments (MOVE) workshop in conjunction with IEEE INFOCOM, pages 103 – 108, 2007.

[39] Richard H. R. Harper. Why people do and don't wear active badges: A case study. *Computer Supported Cooperative Work (CSCW96)*, 4, 1995.

[40] Baik Hoh and Marco Gruteser. Protecting location privacy through path confusion. In *IEEE/CreateNet Intl. Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm)*, September 2005.

[41] Jason I. Hong. *An Architecture for Privacy-Sensitive Ubiquitous Computing.* PhD thesis, University of California at Berkeley, 2005.

[42] Yih-Chun Hu and Helen J. Wang. A framework for location privacy in wireless networks. In *ACM SIGCOMM Asia Workshop*, 2005.

[43] Sotiris Ioannidis, Stelios Sidiroglou, and Angelos D. Keromytis. Privacy as an operating system service. In *1st USENIX Workshop on Hot Topics in Security*, 2006.

[44] Saving lives through advanced vehicle safety technology: Intelligent vehicle initiative final report.

[45] Xiaodong Jiang, Jason I. Hong, and James A. Landay. Approximate information flows: Socially-based modeling of privacy in ubiquitous computing. In *Ubicomp*, 2002.

[46] Eija Kaasinen. User needs for location-aware mobile services. *Personal and Ubiquitous Computing*, 7, 2003.

[47] Pandurang Kamat, Yanyong Zhang, Wade Trappe, and Celal Ozturk. Enhancing source-location privacy in sensor network routing. In *the 25th International Conference on Distributed Computing Systems (ICDCS 2005)*, 2005.

[48] Stephen T. Kent and Lynette I. Millett, editors. *Who Goes There? Authentication Through the Lens of Privacy.* National Academy Press, 2003.

[49] Alfred Kobsa. Privacy-enhanced personalization. *Communication of the ACM*, pages 24 – 33, August 2007.

[50] John Krumm. A survey of computational location privacy. *Personal Ubiquitous Comput.*, 13:391–399, August 2009.

[51] E. Kushilevitz and R. Ostrovsky. Replication is not needed: single database, computationally-private information retrieval. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science*, pages 364–, Washington, DC, USA, 1997. IEEE Computer Society.

[52] Marc Langheinrich. Privacy by design - principles of privacy-aware ubiquitous systems. In *Ubicomp*, 2001.

[53] Philip Levis, Nelson Lee, Matt Welsh, and David Culler. Tossim: Accurate and scalable simulation of entire TinyOS applications. In *the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, 2003.

[54] Xiaodong Lin, Xiaoting Sun, Pin-Pan Ho, and Xuemin Shen. Gsis: A secure and privacy preserving protocol for vehicular communications. *IEEE Transactions on Vehicular Technology*, 56:3442 – 3456, 2007.

[55] Christian Lochert, Björn Scheuermann, and Martin Mauve. Probabilistic aggregation for data dissemination in VANETs. In *Proceedings of the 4th ACM international workshop on Vehicular ad hoc networks*, pages 1–8, 2007.

[56] Joseph Meyerowitz and Romit Roy Choudhury. Hiding stars with fireworks: location privacy through camouflage. In *Proceedings of the 15th annual international conference on Mobile computing and networking*, MobiCom '09, pages 345–356, New York, NY, USA, 2009. ACM.

[57] Wireless client update for Windows XP with Service Pack 2.

[58] Ronald Miller and Qingfeng Huang. An adaptive peer-to-peer collision warning system. In *Vehicular Technology Conference (Spring)*, 2002.

[59] Ulf Moeller, Lance Cottrell, Peter Palfrader, and Len Sassaman. Mixmaster protocol version 2. Technical report, Network Working Group, 2004.

[60] Tamer Nadeem, Sasan Dashtinezhad, Chunyuan Liao, and Liviu Iftode. Trafficview: traffic data dissemination using car-to-car communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 8:6–19, July 2004.

[61] Cheol Oh, Stephen G. Ritchie, and Shin-Ting Jeng. Anonymous vehicle reidentification using heterogeneous detection systems. *IEEE Transactions on Intelligent Transportation Systems*, 8:460–469, September 2007.

[62] Femi Olumofin and Ian Goldberg. Revisiting the computational practicality of private information retrieval. In *Financial Cryptography 2011*, 2011.

[63] Celal Ozturk, Yanyong Zhang, and Wade Trappe. Source-location privacy in energy-constrained sensor network routing. In *the 2004 ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN) in conjunction with ACM Conference on Computer and Communications Security*, October 2004.

[64] Himanshu Pagey, Kien Hua, and Chow-Sing Lin. Caching as privacy enhancing mechanism in location-based services. In *Proceedings of the 2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, MDM '09, pages 238–243, Washington, DC, USA, 2009. IEEE Computer Society.

[65] Leysia Palen and Paul Dourish. Unpacking privacy for a networked world. *CHI Letter*, 5, 2003.

[66] Jeffrey Pang, Ben Greenstein, Ramakrishna Gummadi, Srinivasan Seshan, and David Wetherall. 802.11 user fingerprinting. In *ACM Mobicom 2007*, 2007.

[67] A. Pfitzmann and M. Kohntopp. *Anonymity, Unobservability and Pseudonymity – A Proposal for Terminology.* Springer-Verlag, 2001.

[68] Fabio Picconi, Nishkam Ravi, Marco Gruteser, and Liviu Iftode. Probabilistic validation of aggregated data in vehicular ad-hoc networks. In *Proceedings of the 3rd ACM international workshop on Vehicular ad hoc networks*, pages 76–85, 2006.

[69] Irene Pollach. What's wrong with online privacy policies? *Communication of the ACM*, pages 103 – 108, September 2007.

[70] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *the 6th annual international conference on Mobile computing and networking*, 2000.

[71] Maxim Raya, Adel Aziz, and Jean-Pierre Hubaux. Efficient secure aggregation in VANETs. In *Proceedings of the 3rd ACM international workshop on Vehicular ad hoc networks*, pages 67–75, 2006.

[72] C.L. Robinson, L. Caminiti, D. Caveney, and K. Laberteaux. Efficient coordination and transmission of data for cooperative vehicular safety applications. In *Proceedings of the 3rd ACM international workshop on Vehicular ad hoc networks*, 2006.

[73] Sheldon M. Ross. John Wiley & Sons, Inc, 2nd edition, 1996.

[74] Debashis Saha and Amitava Mukherjee. Pervasive computing: a paradigm for the 21st century. *IEEE Computer*, 36, March 2003.

[75] Mahadev Satyanarayanan. Pervasive computing: vision and challenges. *IEEE Personal Communication*, August 2001.

[76] S. Schechter, T. Parnell, and A. Hartemink. Anonymous authentication of membership in dynamic groups. In *Proceedings of the third International Conference on Financial Data Security and Digital Commerce*, January 1999.

[77] Kewei Sha, Yong Xi, Weisong Shi, Loren Schwiebert, and Tao Zhang. Adaptive privacy-preserving authentication in vehicular networks. In *Proceedings of IEEE International Workshop on Vehicle Communication and Applications*, 2006.

[78] Sanjay Shakkottai. Asymptotics of query strategies over a sensor network. In *IEEE INFOCOM*, 2004.

[79] Aameek Singh, Ling Liu, and Mustaque Ahamad. Privacy analysis and enhancements for data sharing in *nix systems. In *USENIX06*, 2006.

[80] Radu Sion and Bogdan Carbunar. On the practicality of private information retrieval. In *NDSS*. The Internet Society, 2007.

[81] Asim Smailagic and David Kogan. Location sensing and privacy in a context-aware computing environment. *IEEE Personal Communications*, 9, October 2002.

[82] Dawn Xiaodong Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *IEEE Symposium on Security and Privacy*, pages 44–55, 2000.

[83] Karthik K. Srinivasan and Paul P. Jovanis. Determination of number of probe vehicles required for reliable travel time measurement in urban network. *Transportation Research Record*, 1537:15–22, 1996.

[84] Ahren Studer, Mark Luk, and Adrian Perrig. Efficient mechanisms to provide convoy member and vehicle sequence authentication in VANETs. In *Proceedings of 3rd International Conference on Security and Privacy in Communication Networks (SecureComm'07)*, September 2007.

[85] Carlos Sun, Glenn Arr, and Ravi P. Ramachandran. Vehicle reidentification as method for deriving travel time and travel time distributions: Investigation. *Transportation Research Record*, 1826:25–30, 2003.

[86] Carlos C. Sun, Glenn S. Arr, Ravi P. Ramachandran, and Stephen G. Ritchie. Vehicle reidentification using multidetector fusion. *IEEE Transactions on Intelligent Transportation Systems*, 5:155–164, September 2004.

[87] Xiaoting Sun, Xiaodong Lin, and Pin-Pan Ho. Secure vehicular communications based on

group signature and id-based signature scheme. In *Proceedings of International Conference on Communications (ICC)*, 2007.

[88] Vassilios S. Verykios, Elisa Bertino, Igor Nai Fovino, Loredana Parasiliti Provenza, Yucel Saygin, and Yannis Theodoridis. State-of-the-art in privacy preserving data mining. *ACM SIGMOD Record*, 33, March 2004.

[89] Vehicle infrastructure integration, 2006. `http://www.its.dot.gov/vii/docs/vii_factsheet.pdf`.

[90] Ziyuan Wang, Lars Kulik, and Kotagiri Ramamohanarao. Proactive traffic merging strategies for sensor-enabled cars. In *Proceedings of the 4th ACM international workshop on Vehicular ad hoc networks*, 2007.

[91] Mark Weiser. The computer for the 21st century. *Scientific American*, September 1991.

[92] Peter Williams and Radu Sion. Usable PIR. In *NDSS*. The Internet Society, 2008.

[93] Lars Wischhof, André Ebner, and Hermann Rohling. Information dissemination in self-organizing intervehicle networks. *IEEE Transactions on Intelligent Transportation Systems*, 6:90–101, March 2005.

[94] Sergey Yekhanin. Private information retrieval. *Commun. ACM*, 53:68–73, April 2010.

[95] Nickolai Zeldovich, Silas Boyd-Wickizer, Eddie Kohler, and David Mazières. Making information flow explicit in HiStar. In *NSDI06*, 2006.

[96] Feng Zhu, Matt W. Mutka, and Lionel M. Ni. A private, secure, and user-centric information exposure model for service discovery protocols. *IEEE Transactions on Mobile Computing*, 5(4):418–429, 2006.

ABSTRACT

LOCATION PRIVACY IN EMERGING NETWORK-BASED APPLICATIONS

by

YONG XI

December 2012

Advisor: Dr. Loren Schwiebert

Co-Advisor: Dr. Weisong Shi

Major: Computer Science

Degree: Doctor of Philosophy

With the wide spread of computer systems and networks, privacy has become an issue that increasingly attracts attention. In wireless sensor networks, the location of an event source may be subject to unintentional disclosure through traffic analysis by the attacker. In vehicular networks, authentication leaves a trail to tie a driver to a sequence of time and space coordinates. In a cloud-based navigation system, the location information of a sensitive itinerary is disclosed. Those scenarios have shown that privacy protection is a far-reaching problem that could span many different aspects of a computer/network system, especially on a diversified landscape of such systems.

To address privacy protection, we propose to look at the issue from three aspects. First, traffic analysis represents one class of problems. This is because in general, encryption can be applied to protect the information being transmitted but the pattern of transmission is harder to hide due to other constraints. To defeat traffic analysis, it is necessary to identify those constraints and decide the trade-off between them and privacy protection. We have shown that the threat to the source location privacy within a sensor network is directly related with the pattern of the routing protocol. Thus to completely remove that pattern, we propose to use a random walk to defeat the threat.

Second, authentication is generally required to establish the identities of interacting parties in an electronical communication. But it unnecessarily reveals other private information when it is applied to a vehicular network. We propose to introduce tunable anonymity, through both

asymmetric and symmetric encryption primitives, into the authentication process so that it provides k-anonymity. We further extend the scenario to mobile scenarios.

At last, it is often perceived that private information has to be shared in order to obtain certain services. For example, source location and destination location have to be sent for looking up the shortest path between them. We show that it is possible to apply private information retrieval so that a service provider knows for whom it has provided service for accounting purposes, but not the details of the service.

In general, the three aspects represent some basic aspects of privacy issues arising from using computer/network systems. Our approaches, while innovative for the scenarios discussed at hand, are general enough to be applied to similar scenarios.

Keywords: Privacy, Location Privacy, Sensor Network, Anonymity, Shortest Path Routing

AUTOBIOGRAPHICAL STATEMENT

Yong Xi is currently a Ph.D. candidate in the Networking Wireless Sensors Laboratory (NeWS Lab) in the Department of Computer Science at Wayne State University. He received both his MS degree in Computer Science and his BE degree in Computer Science and Engineering from Beihang University, Beijing, China. He is also currently working as a Software Engineer at Amazon.com Inc., Seattle, WA. His areas of research interests include security/privacy and large scale distributed systems. He can be reached at yongxi@wayne.edu.